vMCN: Virtual Mobile Cloud Network for Realizing Scalable, Real-time Cyber Physical Systems

Kiyohide Nakauchi
National Institute of Information and Communications Technology
nakauchi@nict.go.jp

Francesco Bronzino
WINLAB, Rutgers University
bronzino@winlab.rutgers.edu

Yozo Shoji
National Institute of Information and Communications Technology
shoji@nict.go.jp

Ivan Seskar
WINLAB, Rutgers University
seskar@winlab.rutgers.edu

Dipankar Raychaudhuri
WINLAB, Rutgers University
ray@winlab.rutgers.edu

ABSTRACT

This paper presents virtual Mobile Cloud Network (vMCN), an architecture for scalable, real-time Cyber Physical Systems (CPS) based on virtualization-capable network infrastructure and highly distributed edge clouds. Emerging CPS applications running on mobile devices, such as Augmented Reality (AR) based navigation and self-driving cars, have fundamental limitations; (1) the response time over the networks is unmanageable and CPS applications suffer from large response times, especially over shared wireless links; (2) the number of real-virtual object mappings cannot be scaled to the approaching trillion order of magnitude in an unified manner. vMCN addresses these issues by introducing novel network virtualization techniques that exploit the "named object" abstraction provided by a fast and scalable global name service. Coordinating virtualized resources across multiple domains, vMCN supports the distributed edge cloud model by deploying an application aware anycast services that achieve the strict requirements of CPS while still scaling to the expected order of magnitude of devices. An initial vMCN prototype system has been developed using the ORBIT testbed resources and the NICT's virtualization-capable WiFi base station. Experimental results reveal the vMCN can support up to about 94% CPS cycles under the set goal of 100 ms, outperforming the baseline system by almost two times.

1. INTRODUCTION

The emerging network services connecting and controlling machines, vehicles and other objects in the physical world often referred to as the Internet-of-Things (IoT) or cyber physical system (CPS), represent an important set of future requirements for the network research community. Unique technical challenges associated with the IoT and CPS scenarios include scaling the Internet architecture to support a very large number of objects including wireless/mobile devices, information, network addresses, virtual machines, etc., efficient integration of cloud computing services necessary to serve CPS systems, designing appropriate security and trust models, and achieving fast response for real-time/closed-loop applications. Fundamentally new architectural approaches to networking will be required to address the emerging needs of IoT and CPS scenarios.

DCC'16, July 25-28, 2016, Chicago, IL, USA

 ISBN 978-1-4503-4220-9/16/07…$15.00
DOI: http://dx.doi.org/10.1145/2955193.2955201

The target application of this study is mobile real-time CPS, such as Augmented Reality (AR) based navigation and self-driving cars. The first key challenge is scaling to billions or trillions of objects. It could be easily imagined that nation-wide and large-scale AR services are required to handle such very large number of objects. The second key challenge is low latency in applications. For example, glass device based AR applications require less than 100 ms response time, while self-driving cars require 10 ms order of magnitude. Such extreme requirements cannot satisfied by the current mobile network systems.

This paper proposes a virtual mobile cloud network (vMCN) architecture for emerging scalable, real-time CPS applications. The idea is to create a single virtual network (or "slice") for a given cloud service, which provides the illusion of a local network with uniform authentication of devices and services, seamless mobility of devices/users, dynamic migration of BS and cloud resource and state, coordinated with some level of managed wireless resource allocation, and highly distributed edge clouds. Novel network virtualization techniques are introduced that exploit the "named object" abstraction provided by a fast and scalable global name service. The first challenge of scaling to billions or trillions of objects is addressed by the presence of the global name resolution service (GNRS), a key stone of the MobilityFirst (MF) architecture [11] that serves as the networking foundation for vMCN, which is designed to provide fast dynamic binding 100B-1T object names (globally unique identifiers known as GUIDs) and their current network locators. The second key challenge of achieving low latency is achieved by the two following novel mechanisms: first, cloud service addressability and anycast capabilities enabled by the name base routing available in MF and enhanced through virtualization techniques deployed into the network; second, the prioritization of a specific service in wireless access enabled by dynamic assignment and migration of virtualized BS resources supported by the foundation of the MF GUID service layer.

The design of vMCN architecture lays its foundations on the three core technology components and coordination techniques among them: a) GNRS, b) MF-VN, a virtual network framework based on the MobilityFirst architecture that exploits the name-object concept to support clean management of VN resources and enhance application specific requirements through a technique called Applications Specific Routing [1]; finally c) vBS, a virtual network framework in WiFi networks [6, 7]. The initial proof-of-concept prototype of the vMCN was developed based on the integration of the MF-VN prototype on the ORBIT testbed, and the vBS prototype system developed by NICT. Through a set of experiments on the vMCN prototype, the impact on the reduction of CPS response time was evaluated. The experimental results reveal the vMCN can

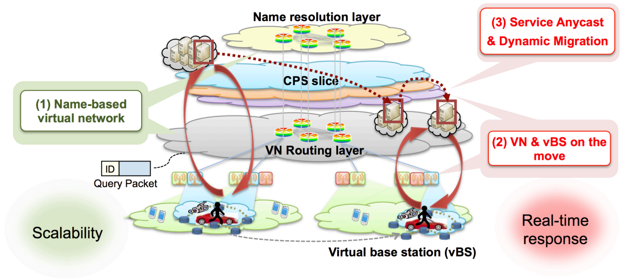

Figure 1: vMCN Architecture Design

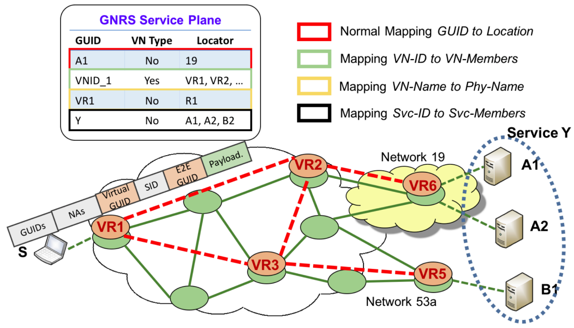

Figure 2: MF-VN Architecture Design and Name Based Definition

support up to about 94% CPS cycles under the set goal of 100 ms, outperforming the baseline system by almost two times.

This paper is structured as follow. Section 2 introduces the proposed vMCN architecture based on the three fundamental technologies: GNRS, MF-VN, and vBS. Section 3 describes the protocol details of the vMCN, especially from the perspective of building an integrated virtual network in vMCN. Section 4 presents the design of the proof-of-concept prototype developed to evaluate the architecture, while Section 5 provides the experimental results on the impact of vMCN for reducing the application-level response time in CPS applications. Finally, Section 6 concludes this paper.

2. VMCN DESIGN

2.1 Overview

The designed architecture exploits two fundamental technologies developed at Rutgers and NICT: a Virtual Network designed on top of the name based Future Internet architecture MobilityFirst (MF-VN) [11, 1] for inter domain support of the applications and the virtual Base Station (vBS) for local domain support [6, 1]. The two components are merged into an architecture that is aimed at supporting a variety of services, including the trillion-order scalability CPS applications with less than 100 ms response time. The comprehensive architecture, as shown in Figure 1, relies on four main characteristics: 1) a fast and scalable global name resolution for user mobility through a Global Name Resolution Service (GNRS), 2) a virtual network with dynamic configuration of wired & wireless resources and inter-domain migration, and 3) a network-assisted service anycast routing service for supporting edge cloud services. A brief description of the components will follow. For further details, please refer to [6, 1].

Though our initial target of the response time is 100 ms, we do not consider this meets the requirements of all the real-time CPS applications. Actually, a self-driving car and precise machine control in manufacturing require 10 ms ans 1 ms order of magnitude, respectively. The importance is controllability of the response time, and we believe the vMCN architecture can be further enhanced to support such ultra low latency by placing edge clouds closer to end devices.

2.2 GNRS: Name Based Services for Inter-Domain Coordination

Separating names (identities) from addresses has been advocated by the research community [5, 11] for quite some time and has inherent benefits in handling mobility and dynamism for one-to-one communication. But they also provide additional advantages by facilitating creation of new service abstractions that can be used to design new and advanced services, like the Virtual Network here described. First, names can be used to represent many different Internet objects; for example, a cell-phone, a person, or a group of devices; the latter perfectly applies in the context of virtual networks helping to define participating resources. Moreover, new logical layers can be integrated within these names; for example, we gain the ability to map logical resources at the VN layer, i.e. virtual routers, to their physical counterparts, i.e. the physical machines hosting them.

In the vMCN architecture, name separation is achieved through the employment of a logically centralized, globally distributed name resolution service associated with name based communications. The Global Name Resolution Service (GNRS) is then a network-wide entity that provides an API for inserting and querying mappings between unique name identifiers (called GUIDs in MF) and a set of values which can include network addresses, other name identifiers and related parameters – e.g., node properties, past locations and more. In spirit, this service is very similar to current Internet's DNS, which has already been effectively applied for new service functions such as load balancing and service replication. As vMCN's design is based on MobilityFirst [11], a clean-sate network architecture for the next-generation mobile network, it employs the DMap [13] implementation as GNRS.

The GNRS has a fundamental role in the vMCN architecture providing three core functions: 1) as vMCN is a derivative of the MF architecture, the GNRS is used to support a hybrid name-address forwarding scheme, in which routing components use availability of both names and addresses in packet headers to perform forwarding decisions (for more details refer to [11, 9]); 2) it provides the required abstractions exploited by the Virtual Network presented next in Section 2.3; 3) it is the central component that allows to implement coordination techniques across the different layers of virtualization. More details about these mechanisms will follow in Section 3

2.3 MF-VN: Virtualization of MobilityFirst-based Networks

The MobilityFirst Virtual Network [1] has been designed to exploit the name object abstraction derived from the name/address separation provided by the GNRS to define a clean and simple logic for defining and managing VNs. Name based communications provide native support of virtual networks as the named-object abstraction makes it possible to define virtual networks and store the corresponding topology directly in the GNRS. Figure 2 shows how such information is stored in the service: first, marked in *green*, a unique name is assigned to the Virtual Network (*VNID_1*) and mapped to the list of participating routers. Such list contains unique names that only belong to the VN logic. These names are then mapped (in *yellow*) to the ones identifying the physical resource, i.e. the router. This indirection allows for a clean separation between logical and physical layer allowing for easier management and maintenance procedures, e.g., migration. Routers can support multiple virtual network policies simply by indexing their routing table to the virtual network identifier. This allows us to operate VN 's on the MF architecture without the need for any additional overlay protocols.

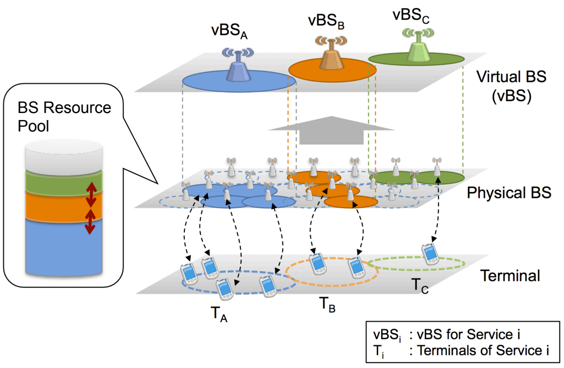

Figure 3: Concept of WiFi Network Virtualization (vBS)

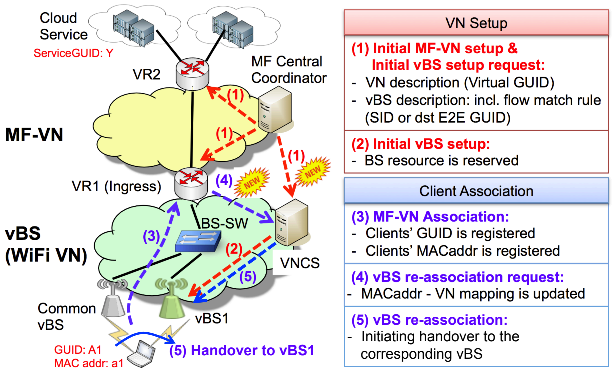

Figure 4: The control planes of MF-VN and vBS are coordinated for building an integrated virtual network

In order to support advanced mobile edge cloud scenarios where the goal is to connect mobile devices to the "best" or "nearest" cloud server the VN design was designed to further support *anycast* delivery services. While basic anycast services have been implemented over the years through the employment of DNS based techniques or through application based overlays, providing it as an integrate abstraction of the networking components would highly benefit system performance, by reducing experienced latency of the system. This is of fundamental importance in highly dynamic environments required by mobile edge clouds where the location of the service often migrates to closely follow moving users. Moreover, the concept of anycast could be further extended to allow different definitions of "best" destination. Recent studies [12, 15] demonstrate how allowing applications to affect routing decisions based on their own metric, e.g. server load for load balancing, can have a positive impact both in terms of application and network performance. A clean-slate design should aim at taking advantage of the opportunity and provide enough flexibility to support coordination across application and network logic.

Again, in this scenario, the name-object abstraction allows network resources to take more informed decisions. Considering the example in Figure 2, the *black* mapping represents how a service name could be be bound to its participants allowing the forwarding fabric to obtain the potential desired destination. With the provided *anycast* service, the client solely needs to specify the name of the service and will be routed to the best destination site. While this abstraction is powerful. In most cases, the definition of the best service replica to select can vary, depending on the nature of the application. For this reason, we use our Virtual Network design to support Application Specific Routing (ASR). ASR allows the application components to proactively provide information regarding the status of the service to the routing components. This information, expressed in the form of a single or multiple metric values, can be integrated into the routing algorithms to take more informed decisions. Further details on how ASR is exploited in the vMCN architecture will be provided in Section 3.

2.4 vBS: WiFi Network Virtualization

Figure 3 shows a basic concept of WiFi network virtualization [6]. WiFi network virtualization is a technique in which physical WiFi network infrastructure resources and physical radio resources can be abstracted and shared by multiple independent and customizable logical (virtual) WiFi networks through isolating each other, and can be considered as an example of wireless network virtualization [10, 14, 2, 3, 16, 6]. In this paper, we call a set of physical WiFi network infrastructure resources and physical radio resources as a BS resource for simplicity. In Figure 3, a set of isolated BS resources is presented as a vBS. A vBS behaves as a logical multi-channel BS organized by multiple physical BS resources. We define a vBS that dedicates all its own BS resources to a target service as a *service-specific* vBS [8]. In the current design, intra-domain migration of the vBS is supported [7].

To make a vBS logically behave as a single BS, a logical layer-2 network spanning across physical BSs that organize the vBS is configured in the backhaul. In addition, all the BSs are configured with the same MAC address in a wireless interface, and hence with the same BSSID. In the same way, the same ESSID are configured at all the BSs. These configurations make it possible to separate BS selection and handover decisions from BSs and terminals and put them together into a centralized controller.

The distinct advantage of the architecture is that all the decisions on BS selection and handover can be fully managed regardless of the differences in the vendor-specific BS selection and handover algorithms implemented in terminals. The association and handover procedures naturally go together with layer-2 routing (re)configurations in the backhaul for the terminals, so that OpenFlow [4] is exploited for this purpose. Reducing handover latency and avoiding packet drops during handover can be also achieved by cooperative and fast layer-2 path reconfiguration.

3. PROTOCOL DETAILS

3.1 Dynamically Configuring a Name-based Virtual Network

The three core technology components previously described represent the foundations required to achieve a scalable architecture to support the real-time CPS. In order to achieve a fully functional architecture a series of coordination techniques are required among those components. The key integration point is a mechanism for building an integrated virtual network by bridging a MF-based virtual network (MF VN) and a corresponding vBS. In this paper we propose a mechanism to coordinate the MF's control plane and the vBS's one for that purpose.

Figure 4 shows a procedure of building an integrated virtual network based on the coordination between the MF resource manager (called central controller in the picture) and the vBS's controller, namely VNCS. The procedure can be divided into two parts: the first part is the VN setup phase (step (1) and (2)) and the latter one is the client association phase (Step (3) – (5)).

In the VN setup phase, the MF resource manager initiates the MF-VN setup by pushing to the participating nodes the definition of requested resources and the unique identifier that characterize the Virtual Network(Step (1)). Note that an original process is added that the coordinator requests the VNCS to create a corresponding vBS using the northbound VNCS API[6]. Details are explained later in Section 4. A unique tag, called Service ID, or

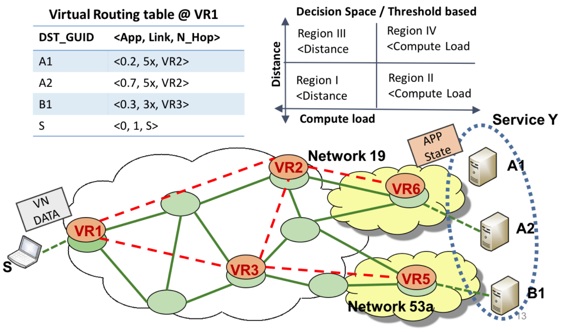

Figure 5: Application Specific Routing (ASR) Concept

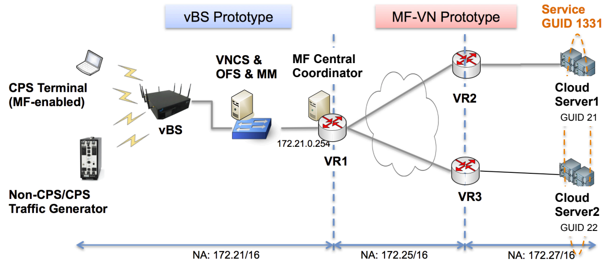

Figure 6: Physical Structure of the vMCN Prototype

alternatively the E2E communication GUIDs, corresponding to the MF-VN is also notified to enable the VNCS to identify the GUID packets of the VN. Then, the VNCS creates the requested service-specific vBS (vBS1) by configuring the physical BSs and the BS switch (BS-SW) in the BS backhaul (Step (2)).

In the client association phase, a MF client is associated with the MF-VN first, and then associated with the service-specific vBS. While the client WiFi interface is active, the client periodically looks for a MF ingress router using broadcast messages. The MF-VN association is completed when the client receives the acknowledgement from the ingress router (Step (3)). The GUID and MAC address of the client are registered in the ingress router. Note that at this step, the MF client is associated with a common vBS instead of the service-specific vBS.

Then the ingress router notifies the VNCS of MAC address – VN mapping information (Step (4)). This information is required for the VNCS to identify which MAC address should be bound with which vBS. The Step (4) is also original in the proposed integration mechanism. Finally, the VNCS initiates the handover of the target client to the service specific vBS (Step (5)).

3.2 Introducing Application Performance Index into Service Anycast

While a name based architecture, like MobilityFirst, is well suited to provide the right abstractions for *Anycast* based services, when in need of meeting strict application performance requirements, additional control over routing decisions might be desired [12, 15]. In order to do so, we developed a concept called Application Specific Routing (ASR). ASR allows routing decisions to be based both on network and application metrics. Consider the scenario shown in Figure 5 where a service *Y* is distributed across three locations *A1*, *A2* and *B1*. When receiving a data packet from client *S*, classic routing protocols would forward such packet based solely on network level information, e.g., bandwidth, latency, hop counts, etc. Through ASR, we provide a framework that allows not only to consider classic L3 metrics, but also application layer ones, such as cloud workload/latency for the edge-cloud scenario.

ASR is implemented as an integrated function of the virtualization layer, whereas participating routers can be configured to support different metrics and forwarding logics. Current ASR's implementation is based on two core operations:

Application Metric Dissemination. When ASR is active, cloud nodes participate in the routing protocol by sending Application State Packets to the edge router they are attached to; the routers then re-distribute the state information across the other participant routers by inclusion of the metrics and related information (e.g., name of the server) into the routing packets exchanged within the Virtual Network.

Forwarding Decisions. The routing information exchanged is used to compute two tables: 1) the classic routing table used for normal networking decisions and 2) a service table that contains the current state (i.e. cloud server load) for the different cloud service locations. When receiving data packets with destination the service, i.e. *Y*, the Application Specific Routing logic is used to select the next hop. The current implementation supports a basic threshold based logic, as shown in Figure 5, where potential destinations are divided into a decision space in which different regions have higher priority based on desired behavior. For example *Region II* could be preferred over *Region III* if network metrics have higher importance over server load; they could be swapped otherwise. Our framework could be easily extended to support different decision logics.

While the current ASR implementation supports basic threshold based routing logic, our design allows for future extensions supporting more complex algorithms based on the collected state information.

4. PROTOTYPING

The initial proof-of-concept vMCN prototype system was developed by integrating the two prototypes of vBS and MF-VN. The MF-VN prototype was developed by implementing the MF-VN logic described in Section 2.3 on the MobilityFirst foundation in MF nodes. In addition, the ASR logic described in Section 3.2 was also implemented, which was running on a MobilityFirst virtual network.

The vBS prototype consists of (1) VNCS (Virtual Network Control Server) that mainly manages vBS creation, radio resource assignment, client association, and handover, (2) virtualization-capable BS, where 22 IEEE 802.11a/b/g/n WiFi modules are equipped and at most 22 WiFi BSs can be run independently at a time, and (3) BS-SW, which is an OpenFlow switch. The BS-SW was implemented using Open vSwitch in this integrated prototype.

Figure 6 shows the physical structure of the vMCN prototype. The MF-VN part consists of three MF routers and two cloud servers. These five servers as well as a GNRS node were built on the ORBIT testbed in the Rutgers University. Then the BS-SW (OFS) is connected to the MF ingress router (VR1). The VNCS and the MF central coordinator were run on the BS-SW and VR1, respectively.

For integrating these two prototypes, the function of dynamically configuring a virtual network among vBS and MF-VN was implemented. Specifically, the function of calling an initial vBS setup request (Step (1) in Figure 4) was implemented in the MF central coordinator. This function is embedded into the existing initial MF-VN setup procedure. The function of calling a vBS re-association request (Step (4) in Figure 4) was implemented in the MF ingress router. This function is invoked just after the MF-VN association procedure.

The above new functions require communications with the VNCS, and a software module for calling the northbound API of the VNCS, namely VNCS API [6], was installed in the MF central coordinator and the ingress router. The VNCS API provides a series of meth-

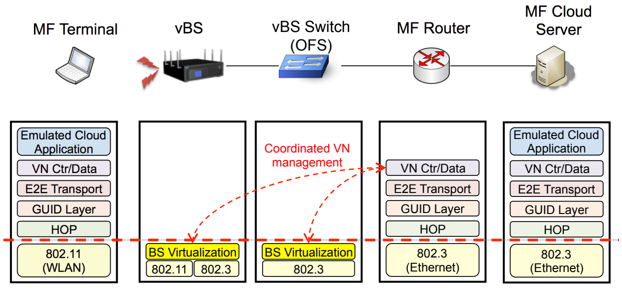

Figure 7: Protocol Stack

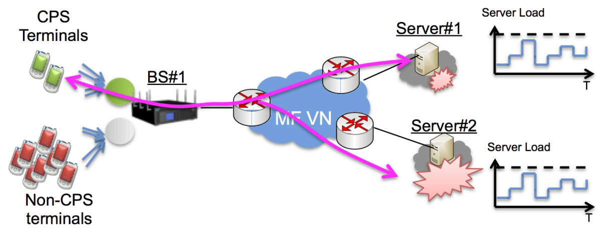

Figure 8: Experimental Scenario

ods, such as SliceAdd, vBSAdd, and vBSAttachBS for creating a vBS. These methods are used in Step (1), and the EnvHandover method is used in Step (4).

Figure 7 shows the protocol stack implemented on the vMCN prototype system. The vBS stack is working as a layer-2 protocol, and a MobilityFirst GUID packet encapsulated by the IEEE 802.3 or IEEE 802.11 header can be routed on the vBS and BS-SW as it is. The interaction between the BS virtualization layer and the VN control/data layer denotes the process of dynamically configuring a virtual network.

5. REDUCTION OF THE CPS RESPONSE TIME

5.1 Experimental Setup

A first batch of results showcasing the potential of the vMCN architecture has been obtained by experiments on the prototype system. Figure 8 shows the basic experimental scenario, which can evaluate both the individual effects of the vBS on vWiFi and ASR on MF-VN and the marginal one.

We generated loosed-loop (round-trip) UDP traffic and one-way UDP traffic at terminals to emulate a real-time CPS service and a non-CPS best-effort one, respectively. The CPS-specific vBS and the best-effort common vBS on the virtualization capable BS (vcBS), respectively, and a physical BS configured with IEEE 802.11 n/a mode and 65 Mbps transmission rate is assigned to each vBS. The channel 36 and 48 in the 5 GHz band are assigned to these vBS, respectively. When these vBSs are activated, CPS and non-CPS traffic are completely isolated, and the CPS response time can be reduced. On the other hand, without the vBSs, i.e. in the normal WiFi mode, the interference causes the increase of the response time.

In the cloud servers, dynamic server load are configured. Every 10 seconds, each server randomly chooses the server load from the preconfigured parameter set {0.2, 0.4, 0.6, 0.8}, and linearly increased latency of {20, 40, 60, 80} ms is injected before responding to the received CPS data unit. The server load is announced every 2 seconds to the MF routers, and then the ASR routing table is updated accordingly. When the ASR is activated on the MF-VN and servers, the ASR routes the CPS traffic to the less-loaded server, and the CPS response time can be reduced. On the other hand, without the ASR, i.e. in the normal GNRS mode, the CPS traffic are always routed to Server#1, and the response time is significantly affected during the high server load period.

To evaluate the application-level CPS performance instead of the packet-level one, we generate a large size data unit assuming a picture frame. For example, in the case of 50 KB data unit and 1500 Byte MTU, a packet train including 37 MF-formatted packets is generated. We generated a CPS data unit every second at each CPS terminal. The response time is defined as the time from the generation of a data unit to the reception of the last packet of the packet train. If any packet in the train is lost, we consider the data unit is lost.

We setup three CPS terminals and 12 non-CPS ones per physical BS. We configure two physical BSs, and 30 terminals are configured in total. This means that when vBSs are activated, the CPS-specific vBS accommodates 6 CPS terminals and the remaining 24 non-CPS terminals are accommodated by the non-CPS vBS, when vBSs are activated. On the other hand, without vBSs, each physical BS accommodates 3 CPS terminals and 12 non-CPS ones. We generated the non-CPS traffic with 100KB data unit every 1 second at each non-CPS terminal, and the load offered by 12 non-CPS terminals is 9.65 Mbps. We implemented the CPS traffic generator, named ßmfping, on a Linux laptop to measure the CPS response time. The other CPS and non-CPS terminals are configured on the IXIA WiFi terminal emulator. All the terminals are connected to the vcBS using coaxial cables to eliminate the interference from the other wireless systems using the ISM band.

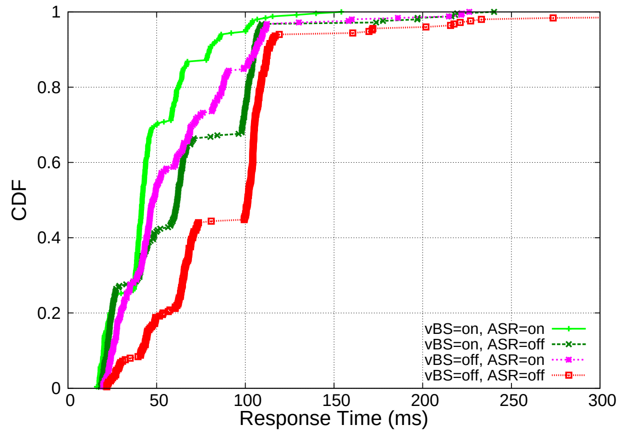

Figure 9: CDF of CPS Response Time (Data Unit Size = 25KB)

5.2 Experimental Results

Figure 9 shows the cumulative distribution function (CDF) of CPS response time. 300 data units with 25KB size are generated in this experiment. Apparently the combination of vBS and ASR outperforms the other three cases. As we target the response time less than 100ms, we focus on how many data units meet the requirement. In the case of vMCN, i.e. (vBS, ASR) = (on, on), 94% data units achieves less than 100ms response time. On the other hand, in the case of (vBS, ASR) = (off, on), (on, off), and (off, off), the value decreases to 85%, 74%, and 46%, respectively. These results show ASR has larger impact to lift up the CDF line, We can conclude the vMCN can support up to 94% CPS cycles under the set goal of 100 msec, and outperforms the baseline system by almost two times.

The specific percentile response time is also an important performance index to evaluate a real-time system. We evaluate the performance in the congested WiFi environment, we determines to

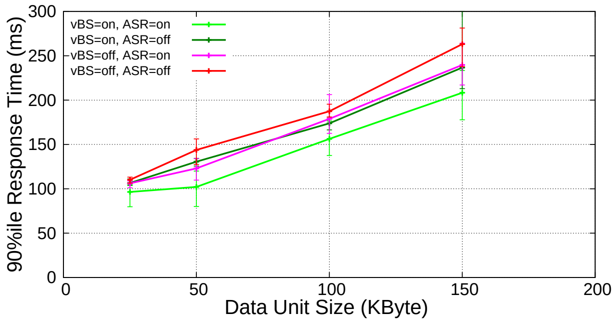

Figure 10: 90 Percentile Response Time

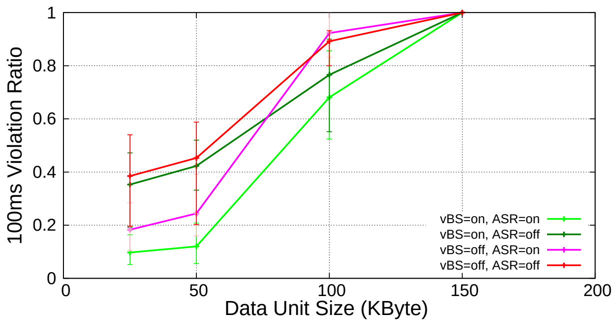

Figure 11: 100ms Delay Violation Ratio

evaluate 90 percentile response time. In the case of (vBS, ASR) = (on, on), the 90 percentile response time is 80 ms. On the other hand, in the case of (vBS, ASR) = (off, on), (on, off), and (off, off), the value increases to 106 ms, 105 ms, and 113 ms, respectively. These results show vBS and ASR have the same level of impact. We can conclude the vMCN can achieve less than 100 msec for 90 percentile response time for 25KB CPS data units.

Figure 10 shows the impact of data unit size on the 90 percentile response time. We measured 10 times (obtained 10 sets of 300 samples) for each parameter set. The average values with the range between the minimum and maximum ones are plotted in the figure. In all the cased, the larger data unit size has larger impact on the response time. The slope on each graph is almost the same, and the effects are common for all the cases. We can conclude the it is difficult for vMCN to keep the 90 percentile response time less than 100 ms for data units with larger than 50KB. Note that the non-CPS traffic is static, and the ratio of CPS traffic becomes larger for larger CPS data unit size. For example, the load offered by three CPS terminals is 4.0 Mbps when the CPS data unit size is 150 KB.

Figure 11 shows the impact of data unit size on the 100 ms violation ratio. The 100 ms violation ratio is expressed as $1 - \frac{N_{success}}{N_{all}}$, where N_{all} and $N_{success}$ denote the total number of generated data units and the total number of data units that meet the response time of less than 100 ms, respectively. In the case of (vBS, ASR) = (on, on), the 100 ms violation ratio is 0.10 and 0.12 for 25 KB and 50 KB data units, respectively. When the unit size is 50 KB, the 100 ms violation ratio in the case of (vBS, ASR) = (off, on), (on, off), and (off, off) is 0.24, 0.42, and 0.45, respectively. When the unit size is larger than 100 KB, the ratio significantly increases. We can conclude the it is difficult for vMCN to keep the 100 ms violation ratio less than 0.1 for data units with larger than 50KB.

6. CONCLUSIONS

This paper presented virtual Mobile Cloud Network (vMCN), an architecture for scalable, real-time Cyber Physical Systems (CPS) based on virtualization-capable network infrastructure and highly distributed edge clouds. Emerging CPS applications running on mobile devices, such as Augmented Reality (AR) based navigation and self-driving cars, have fundamental limitations; (1) the response time over the networks is unmanageable and CPS applications suffer from large response times, especially over shared wireless links; (2) the number of real-virtual object mappings cannot be scaled to the approaching trillion order of magnitude in an unified manner. vMCN addressed these issues by introducing novel network virtualization techniques that exploit the "named object" abstraction provided by a fast and scalable global name service. Coordinating virtualized resources across multiple domains, vMCN supports the distributed edge cloud model by deploying an application aware anycast services that achieve the strict requirements of CPS while still scaling to the expected order of magnitude of devices. An initial vMCN prototype system has been developed using the ORBIT testbed resources and the NICT's virtualization-capable WiFi base station. Experimental results revealed the vMCN can support up to about 94% CPS cycles under the set goal of 100 ms, outperforming the baseline system by almost two times.

7. REFERENCES

[1] A. Babu. Design and implementation of a virtual networking framework for the mobilityfirst internet future architecture. Master's thesis, Rutgers University, 2015.

[2] F. Fu and U. C. Kozat. Stochastic Game for Wireless Network Virtualization. *IEEE/ACM Transactions on Networking*, 21(1), February 2013.

[3] C. Liang and F. R. Yu. Wireless Network Virtualization: A Survey, Some Research Issues and Challenges. *IEEE Comm. Surveys and Tutorials*, 16(3), July 2014.

[4] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner. OpenFlow: Enabling Innovation in Campus Networks. *ACM SIGCOMM Computer Communications Review*, 38(2), April 2008.

[5] R. Moskowitz, P. Nikander, P. Jokela, and T. Henderson. Host identity protocol. *IETF RFC 5201*, 2008.

[6] K. Nakauchi and Y. Shoji. WiFi Network Virtualization to Control the Connectivity of a Target Service. *IEEE Transactions on Network and Service Management*, 12(2):308–319, June 2015.

[7] K. Nakauchi and Y. Shoji. vBS on the Move: Migrating a Virtual Network for Nomadic Mobility in WiFi Networks. *Proc. MobiWorld '16*, April 2016.

[8] K. Nakauchi, Y. Shoji, M. Ito, Z. Lei, Y. Kitatsuji, and H. Yokota. Bring Your Own Network – Design and Implementation of a Virtualized WiFi Network –. *Proc. IEEE CCNC'14*, January 2014.

[9] S. C. Nelson, G. Bhanage, and D. Raychaudhuri. Gstar: generalized storage-aware routing for mobilityfirst in the future mobile internet. In *Proceedings of the sixth international workshop on MobiArch*, pages 19–24. ACM, 2011.

[10] S. Paul and S. Seshan. Virtualization and Slicing of Wireless Networks. *GENI Design Document 06-17, GENI Wireless Working Group*, September 2006.

[11] D. Raychaudhuri, K. Nagaraja, and A. Venkataramani. Mobilityfirst: a robust and trustworthy mobility-centric architecture for the future internet. *ACM SIGMOBILE Mobile Computing and Communications Review*, 16(3):2–13, 2012.

[12] V. Valancius, N. Feamster, J. Rexford, and A. Nakao. Wide-area route control for distributed services. In *USENIX Annual Technical Conference*, 2010.

[13] T. Vu, A. Baid, Y. Zhang, T. D. Nguyen, J. Fukuyama, R. P. Martin, and D. Raychaudhuri. Dmap: A shared hosting scheme for dynamic identifier to locator mappings in the global internet. In *Distributed Computing Systems (ICDCS), 2012 IEEE 32nd International Conference on*, pages 698–707. IEEE, 2012.

[14] X. Wang, P. Krishnamurthy, and D. Tipper. Wireless Network Virtualization. *Proc. ICNC '13*, January 2013.

[15] X. Wu and J. Griffioen. Supporting application-based route selection. In *Computer Communication and Networks (ICCCN), 2014 23rd International Conference on*, pages 1–8. IEEE, 2014.

[16] M. Yang, Y. Li, D. Jin, L. Zeng, X. Wu, and A. V. Vasilakos. Software-Defined and Virtualized Future Mobile and Wireless Networks: A Survey. *Mobile Networks and Applications*, September 2014.