A 6LoWPAN IoT Platform on the Global Internet

Zengxu Yang
Tufts Wireless Lab
Department of Electric and Computer Engineering
Tufts University
zengxu.yang@tufts.edu

C. Hwa Chang
Tufts Wireless Lab
Department of Electric and Computer Engineering
Tufts University
chorng.chang@tufts.edu

Abstract

This paper describes a platform under development that runs globally interconnected IP IoT networks based on 6LoWPAN through an IPv4/IPv6 mixed environment. Border routers running 6LBR on Linux are used to interface constrained nodes running Contiki with external IPv6 networks. 6in4 tunnels are used to encapsulate IPv6 packets inside IPv4 packets where native IPv6 networks are not available and the number of IPv4 addresses are severely restricted. All the constrained nodes are assigned global IPv6 addresses and are globally routable. End-to-end connectivity is achieved by eliminating NAT in IPv6. Firewall guidelines to make this platform work are also discussed. Configurations and testing results of our 6LoWPAN platform testbed are described. Testing results show that our platform meets design goals and can be used for further IoT development.

1 Introduction

The Internet of Things (IoT), just like the Internet, has the potential to greatly change our lives. Wirelessly connected networks formed by a vast amount of low power, low cost, small devices with various sensors and actuators that run on batteries or even harvest energy from the environment have huge current and potential applications[7]. An IoT system that uses IPv6 based network protocols for all its devices and is globally interconnected, seamlessly integrating with the current Internet and Web services, working on both IPv4 and IPv6 networks, is appealing to both IoT users and developers. Such an IoT system being developed by Tufts Wireless Lab consists of two platforms: an 6LoWPAN platform and a Web platform[23]. This paper describes the globally interconnected 6LoWPAN platform, which connects to the Web platform that processes data from 6LoWPANs. Our IoT system is shown in Figure 1.

International Conference on Embedded Wireless
Systems and Networks (EWSN) 2019
25–27 February, Beijing, China

ISBN: 978-0-9949886-3-8

1.1 IPv6 and 6LoWPAN

IEEE 802.15.4 is a low power, low speed wireless personal area network (WPAN) standard that has a raw data rate up to 250 kbit/s. It is widely used in IoT systems[3, 4, 5, 6]. There are two different categories of constrained IoT networks, Non-IP or IP. 6LoWPAN, an IoT IPv6 adaptation layer, runs on top of IEEE 802.15.4. IPv6 was chosen to be the fabric of IP IoT network because IPv6 is more suitable then IPv4 for IoT systems[22].

1.2 Contiki

Contiki is a BSD licensed open source, lightweight operating system that implements 6LoWPAN using uIP on tiny, severely resource constrained nodes with IEEE 802.15.4 wireless communication capabilities, with memory as low as around 20 KB RAM and 100 KB ROM. It is implemented in the C language and has been ported to a wide range of devices on a variety of architectures, like chips based on the Atmel AVR, MSP430, and ARM Cortex-M3 architectures. Contiki supports dynamic loading and replacement of individual programs and services. It is built around an event-driven kernel but provides optional preemptive multithreading that can be applied to individual processes[14].

By using a duty cycling protocol called ContikiMAC, Contiki achieves ultra low power consumption by turning off the wireless transceivers 99% of the time while still being able to communicate[13].

1.3 CoAP

Constrained Application Protocol (CoAP) was developed as a lightweight HTTP on 6LoWPAN. To work on constrained devices, CoAP uses UDP with its own message based retransmission mechanism instead of TCP. CoAP keeps REST architecture yet requires much less resources. CoAP also works with security measures like DTLS, just like HTTP works with TLS[8, 20]. Thus CoAP was chosen as the application protocol for our platform.

2 6LoWPAN Platform

2.1 Border Router

The Beaglebone Black (BBB) is a small, low power, low cost embedded computer capable of running Linux. It comes with the TI Sitara AM3358BZCZ100, a 1GHz ARM Cortex-A8 processor, 512 MB DDR3 SDRAM, an 10/100 Mb Ethernet RJ45 port, a USB host port and a USB client port, GPIO ports, a MicroSD connector, and others[11].

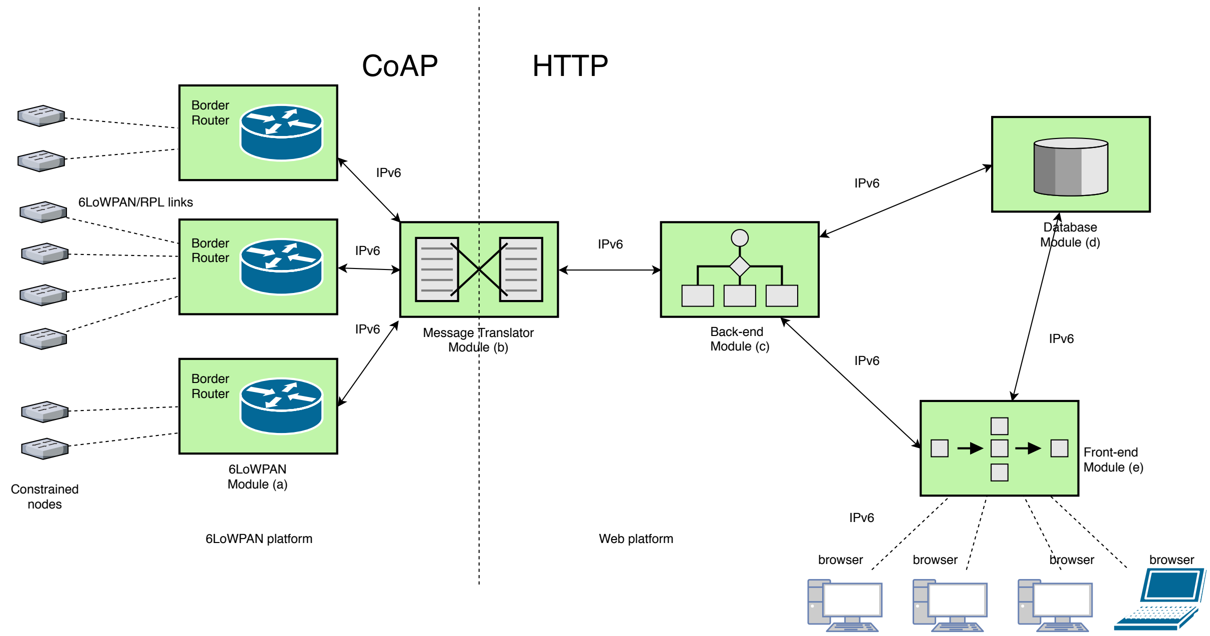

Figure 1. IoT system block diagram

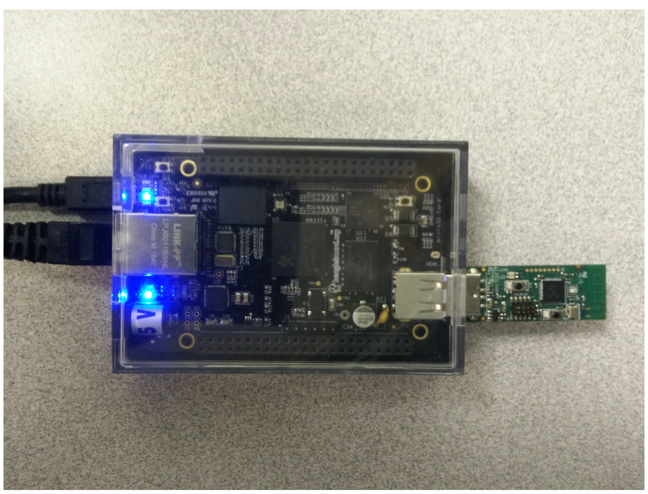

Figure 2. Border router setup

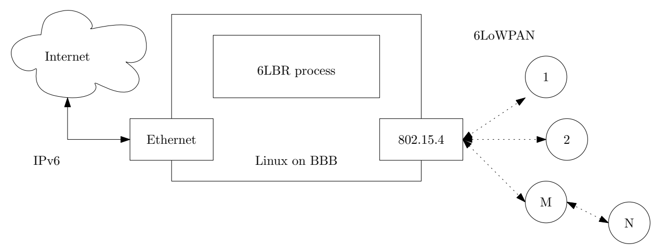

Figure 3. Border router block diagram

Debian GNU/Linux is a community supported Linux distribution that supports major architectures like X86, X86-64, ARM, and MIPS. We use the *armhf* port of Debian 9 to run on the TI Sitara AM3358BZCZ100 processor. The Debian 9 IoT image is flashed to a 4 GB Micro-SD card as the OS on BBB. The IoT image does not contain graphical components, thus reducing its size and making it suited to IoT devices.

A TI CC2531EMK USB dongle is attached to BBB. TI CC2531EMK USB dongle supports IEEE 802.15.4/ZigBee and can be used as a packet sniffer. It is reflashed to support ContikiMAC and communicate with Contiki nodes. The setup of the border router is shown in Figure 2.

CETIC-6LBR[1] is a Contiki based application but it runs as a Linux daemon process to be the 6LoWPAN border routing service. As shown in Figure 3, the border router works as a IP layer gateway to convert between IPv6 packets and 6LoWPAN packets. No application layer protocol translation is needed for 6LoWPAN. On our platform, both sides use CoAP as the application layer protocol, which greatly simplifies the gateway design compared to non-IP protocols like ZigBee.

We use 6LBR version 1.5.0 for our 6LoWPAN platform. 6LBR can be configured to support different modes. The most commonly used ones are Router Mode and Smart Bridge Mode.

In Router Mode, the border router acts as an IPv6 router, interconnecting two different IPv6 subnets. It assigns different IPv6 prefixes to Ethernet port and the IEEE 802.15.4 port. As an IPv6 router, it sends out Router Advertisement (RA) messages to all the neighboring devices, including the sensor nodes. This mode is mostly used as a standalone router that does not need to be globally routed.

As shown in Figure 4, in Smart Bridge Mode the border router acts as an IPv6 NDP proxy on the Ethernet side, accepting RA messages from an upstream IPv6 router and

[1]CETIC stands for Center of Excellence in Information and Communication Technologies. `https://www.cetic.be/` 6LBR stands for 6LoWPAN Border Router.

using the acquired IPv6 prefix and the IEEE 802.15.4 link layer addresses to do IPv6 address stateless autoconfiguration on the 6LoWPAN side. The Ethernet side and the IEEE 802.15.4 side use the same IPv6 prefix.

Our platform uses Smart Bridge Mode to get a globally routable IPv6 prefix from the RA messages sent from an upstream IPv6 router. In case of lacking an IPv6 environment, a 6in4 tunnel and a Linux router that supports tunneling is used as an upstream IPv6 router. The tunnel is transparent to the border router as a Linux router serves as an endpoint of the tunnel and the link between the border router and the Linux router is IPv6.

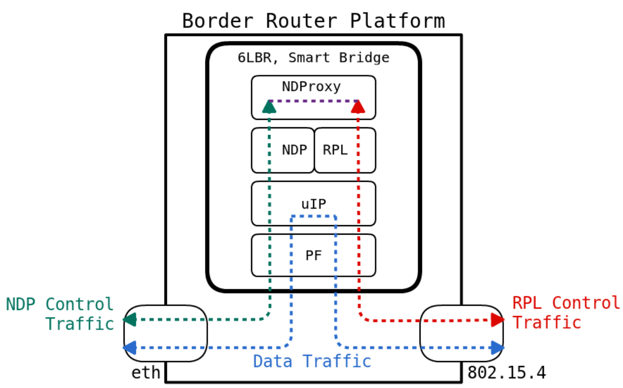

Figure 4. 6LBR Smart Bridge Mode

6LBR includes a web server for configuration and displaying border router information. After successfully running 6LBR on BBB, we visit the IPv6 address of 6LBR from a browser and get the border router Web page, as in Figure 5.

2.2 SensorTag Nodes

The TI SimpleLink SensorTag is an IoT prototyping platform that has numerous sensors and supports major IoT network protocols. CC2650STK SensorTag is based on the ultra-low power CC2650 wireless MCU and supports the development of ZigBee, 6LoWPAN, and BLE. The CC2650 MCU has 20 KB RAM and 128 KB Flash, thus it is class 2 constrained device[9]. CC2650STK contains 10 sensors, including support for light, digital microphone, magnetic sensor, humidity, pressure, accelerometer, gyroscope, magnetometer, object temperature, and ambient temperature[1]. We use CC2650STK as our 6LoWPAN sensor nodes in our platform.

2.3 IPv6 Tunneling

IPv4 based Internet is gradually transitioning to IPv6 as more and more ISPs are providing native IPv6 support to their customers. As the next generation protocol for the Internet, IPv6 is superior to IPv4 in multiple ways, so conversion from IPv4 to IPv6 is desirable. Some of the most important benefits of IPv6 are[12]:

- **Expanded Addressing Capabilities**
  IPv6 increases the IP address size from 32 bits to 128 bits. This greatly increases the total number of IP addresses.

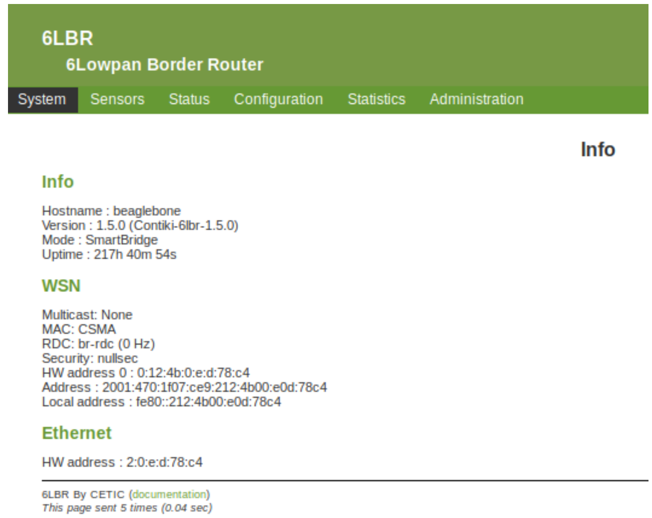

Figure 5. Border router Web page

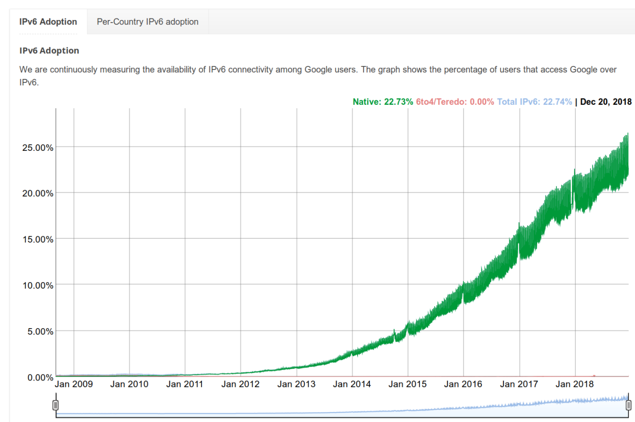

Figure 6. Google IPv6 client statistics

- **Header Format Simplification**
  This makes header processing in routers much more efficient.

- **Authentication and Privacy Capabilities**
  Extensions to support authentication, data integrity, and (optional) data confidentiality are specified for IPv6.

Nonetheless, conversion from IPv4 to IPv6 cannot be achieved in a short time because IPv4 has been used for decades and still dominates the Internet. Even though IPv4 faces a variety of problems like IP address exhaustion and routing table explosion, workarounds like Classless Inter-Domain Routing (CIDR) and Network Address Translation (NAT) keep it working relatively well. To transition from IPv4 to IPv6, millions of legacy network devices need to be upgraded and it could be costly. Coexistence of IPv4 and IPv6 networks will be a reality in the foreseeable future. Currently IPv4 still dominates the Internet, as shown by Goolge IPv6 client statistics in Figure 6[2].

To make transitioning from IPv4 to IPv6 easier, several transitioning schemes have been developed. Dual stack and tunneling are most widely used transitioning schemes[19].

As a 6LoWPAN platform that aims to work on the global Internet, we cope this problem with IPv4/IPv6 dual stack in Linux and 6in4 tunnel technology.

We use the free 6in4 tunnel broker service[15] provided by Hurricane Electric for our 6LoWPAN platform. 6in4 tunnels are implemented by encapsulating IPv6 packets inside IPv4 packets. In the IPv4 header, the protocol field is set to be 41.

3 6LoWPAN Platform Testbed

3.1 Testbed Overview

The configuration of our platform testbed is shown in Figure 7.

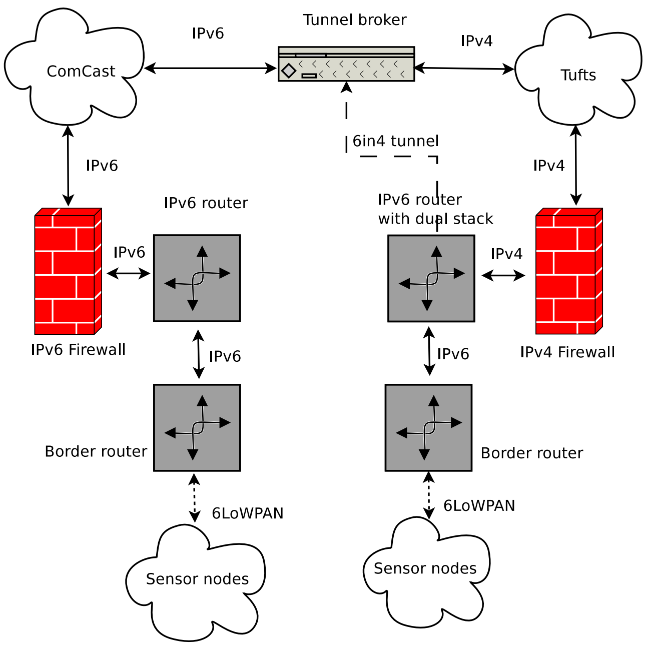

Figure 7. 6LoWPAN platform testbed

We set up two interconnected 6LoWPANs. One sits on the Comcast Xfinity ISP with native IPv6 support, the other one sits on the Tufts campus network with only IPv4 support where we use a 6in4 tunnel to get IPv6 support.

3.2 6LoWPAN with Native IPv6

Comcast Xfinity cable Internet assigns a /64 IPv6 prefix to its residential users in some areas of the United States by default, but a /60 IPv6 prefix can also be requested on demand. We use a WD My Net N750 wireless router as our Comcast IPv6 router. As the factory firmware does not support IPv6 well, we reflashed it with OpenWrt firmware. OpenWrt is an embedded Linux distribution that is used as an alternative router firmware. OpenWrt has advanced router functions that most consumer routers lack. By using a package system, it is highly customizable, adding any features the user demands while still keeping a small code size[16]. By using the Smart Bridge Mode of 6LBR in our border router, we get the /64 IPv6 prefix assigned by Comcast ISP router using DHCPv6 Prefix Delegation protocol and the border router acts as an NDP proxy to forward the IPv6 prefix to 6LoWPAN sensor nodes[21]. The border router and all the 6LoWPAN sensor nodes get globally routable addresses and we verified it by pinging nodes remotely from a IPv6 server in Australia.

3.3 6LoWPAN with 6in4 Tunnel

Another 6LoWPAN sits on the Tufts campus network. Tufts campus network is currently an IPv4 only network. The numbers of globally routable IPv4 addresses we can acquire from the Tufts campus network are very limited. Most of the Tufts campus network users use NAT for Internet access. A one-to-one mapping between IPv4 and IPv6 on our SensorTag nodes is out of the question. While we could just use the Router Mode of 6LBR in our border to get a usable IPv6 environment for 6LoWPAN, as experimented in some previous works[10, 17], this configuration is not globally routable and thus cannot be directly connected to our other globally interconnected 6LoWPAN on the Comcast Xfinity network. We decided to use Smart Bridge Mode of 6LBR in our border router and use the Hurricane Electric IPv6 tunnel broker service to set up a 6in4 tunnel instead. The tunnel broker assigns a /64 IPv6 prefix to us and it is more than enough for our 6LoWPAN platform testbed, while only one public IPv4 address is needed. A Linux computer with IPv4/IPv6 dual stack and two Ethernet ports is used as the IPv6 router. One Ethernet port is connected to the 6LoWPAN border router with an IPv6 link, the other Ethernet port is connected to the upstream IPv4 router on the Tufts campus network.

3.4 Firewall Configuration

Firewalls are used both on the Comcast network and the Tufts network for network security considerations. By default, both firewalls prohibit any inbound connections. We configured the firewalls to unblock incoming traffic of ICMPv6 and incoming traffic on port 22, 80, 443 with TCP, and 5683 with TCP/UDP to allow ping, SSH, HTTP, HTTPS, and CoAP connections. We also allow IPv4 protocol 41 traffic on the IPv4 firewall to allow 6in4 tunneling on the Tufts IPv4 network.

3.5 IPv6 Router with Dual Stack Configuration

Modern Linux supports IPv4/IPv6 dual stack with 6in4 tunnel. We use a PC installed with Debian GNU/Linux 9 serving as an IPv6 router for our 6LoWPAN platform. We add the following lines in `/etc/network/interface` file:

```
auto hurricane0
iface hurricane0 inet6 v4tunnel
  address 2001:470:1f06:ce9::2
  netmask 64
  endpoint 209.51.161.14
  local 130.xx.xx.xxx
  gateway 2001:470:1f06:ce9::1
  ttl 255
  up ip link set mtu 1280 dev $IFACE
  up route -6 add 2001:470:1f07:ce9::/64 eno1
```

where `2001:470:1f06:ce9::2` is the tunnel endpoint IPv6 address used by our IPv6 router, `2001:470:1f06:ce9::1` is the tunnel broker gateway IPv6 address, `209.51.161.14` is the tunnel broker endpoint IPv4 address, `130.xx.xx.xxx` is the local IPv4 address, and `2001:470:1f07:ce9::/64` is the IPv6 prefix provided

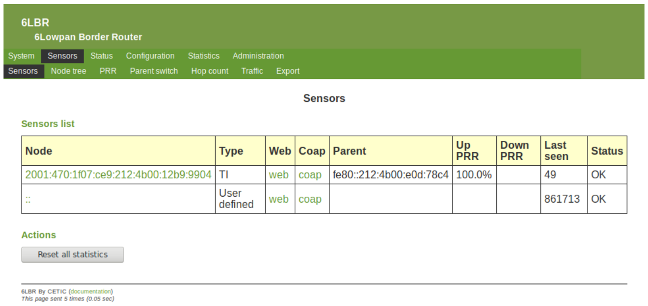

Figure 8. Node information

Figure 9. Programming CC2650STK SensorTag

by the Hurricane Electric tunnel broker that can be used for 6LoWPAN IPv6 address stateless autoconfiguration.

A new network interface is created with the endpoint tunnel IPv6 address `2001:470:1f06:ce9::2`.

```
hurricane0: flags=209<UP,POINTOPOINT,RUNNING,NOARP>  mtu 1280
inet6 2001:470:1f06:ce9::2  prefixlen 64  scopeid 0x0<global>
inet6 fe80::8240:17e0  prefixlen 64  scopeid 0x20<link>
sit  txqueuelen 1  (IPv6-in-IPv4)
```

The router advertisement daemon (radvd) process runs on our Linux IPv6 router to delegate the `2001:470:1f07:ce9::/64` IPv6 prefix acquired from the tunnel broker and uses NDP to configure all 6LoWPAN nodes, as shown in Figure 8.

3.6 SensorTag Nodes Configuration

Contiki images with CoAP support are compiled from Contiki source code and flashed to the CC2650STK SensorTag nodes using TI SmartFlash utility and the TI SimpleLink SensorTag DevPack programmer, as shown in Figure 9. As CC2650STK is a class 1 constrained device, we decide to use CoAP instead of HTTP for application protocol for better IoT resource, power, and traffic utilizations.

4 Testing Results

Both 6LoWPANs are set up successfully and testings are performed to check whether they work properly. Here in Section 4 we demonstrate the testing results from the 6LoWPAN on the Tufts campus network. The one on the Comcast Xfinity network has similar results.

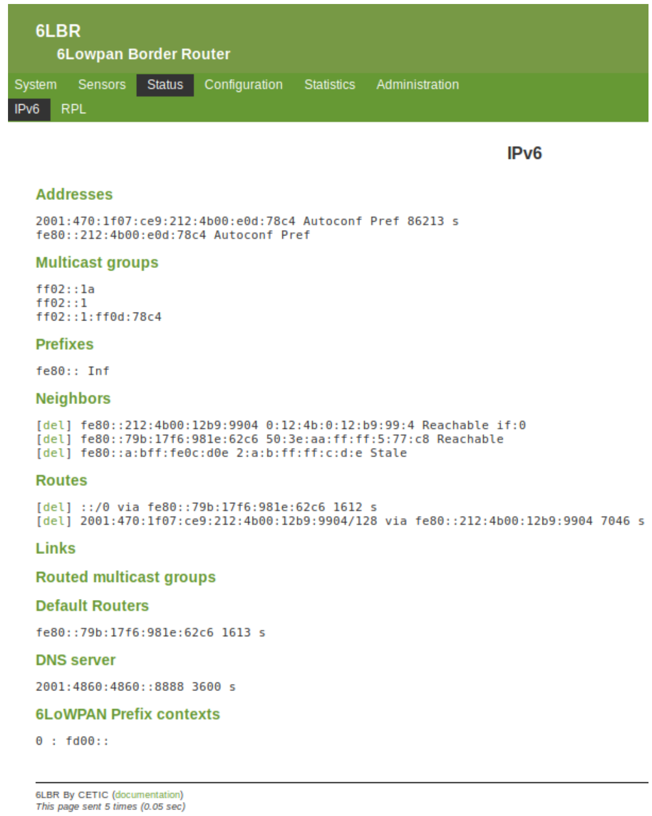

Figure 10. 6LoWPAN status

4.1 6LoWPAN Status

The status of our 6LoWPAN is shown in Figure 10. This shows that the globally routable prefix `2001:470:1f07:ce9::/64` is used in 6LoWPAN. IPv6 address autoconfiguration in 6LoWPAN is successful.

4.2 Global End-to-end Connectivity

A major goal of our 6LoWPAN platform is global end-to-end connectivity. We want our platform to be truly global, such that it can be visited from anywhere with an IPv6 connection. We tested the global end-to-end connectivity by using an IPv6 testing server provided by *IPv6 Now* in Australia[2] to ping our border router and SensorTag nodes. Figure 11 shows the result of pinging one SensorTag node from Australia. The result shows that our 6LoWPANs can be directly visited globally and a global end-to-end connectivity is achieved.

4.3 CoAP Testing Results

Our 6LoWPAN platform uses CoAP as our application protocol. CoAP is used to communicate with Web platform to globally read sensor information and control actuators from SensorTag nodes. We use Copper (Cu) extension of Firefox to test CoAP functionality[18]. As in Figure 12, all the RESTful methods can be used in CoAP and we are able to get sensor readings remotely by using the GET method.

[2] `http://www.ipv6now.com.au`

Figure 11. Pinging SensorTag node globally

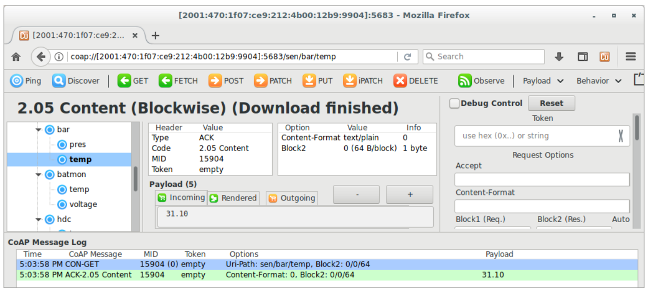

Figure 12. CoAP testing results

5 Conclusion

Our 6LoWPAN platform design and testbed implementation demonstrates a low cost, open source 6LoWPAN IoT network platform that is suited to IoT research and product prototyping. Testing results show that our 6LoWPAN is a truly global IoT platform capable of working in both IPv4 and IPv6 networks. This is particularly important as native IPv6 networks are still not widely available and 6LoWPAN IoT network relies on IPv6 to function properly. By using 6in4 tunneling technology, we get globally routable IPv6 addresses for our 6LoWPANs anywhere IPv4 networks are available, achieving end-to-end connectivity, and successful CoAP based sensor and actuator operations. Such an IoT platform is invaluable for further research and application development, for example, optimization, security research, Web IoT platform integration, which are the subjects of our future work.

6 References

[1] CC2650stk SimpleLink™ Bluetooth low energy Multi-standard SensorTag. https://www.ti.com/store/ti/en/p/product/?p=CC2650STK.

[2] IPv6 – Google. https://www.google.com/intl/en/ipv6/statistics.html.

[3] IEEE Standard for Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks Specific Requirements Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs). *IEEE Std 802.15.4-2003*, pages 1–670, 2003.

[4] IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (WPANs). *IEEE Std 802.15.4-2006 (Revision of IEEE Std 802.15.4-2003)*, pages 1–320, Sept. 2006.

[5] IEEE Standard for Local and metropolitan area networks–Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs). *IEEE Std 802.15.4-2011 (Revision of IEEE Std 802.15.4-2006)*, pages 1–314, Sept. 2011.

[6] IEEE Standard for Low-Rate Wireless Networks. *IEEE Std 802.15.4-2015 (Revision of IEEE Std 802.15.4-2011)*, pages 1–709, Apr. 2016.

[7] L. Atzori, A. Iera, and G. Morabito. The Internet of Things: A survey. *Computer Networks*, 54(15):2787–2805, Oct. 2010.

[8] C. Bormann, A. P. Castellani, and Z. Shelby. CoAP: An Application Protocol for Billions of Tiny Internet Nodes. *IEEE Internet Computing*, 16(2):62–67, Mar. 2012.

[9] C. Bormann, M. Ersue, and A. Keranen. Terminology for Constrained-Node Networks. Technical Report RFC7228, RFC Editor, May 2014.

[10] B. Christensen. BuildingaNetworkfortheInternetofThingswiththeConstrainedApplicationProtocol. page 61.

[11] G. Coley. BeagleBone Black System Reference Manual. page 126.

[12] S. Deering and R. Hinden. Internet protocol, version 6 (IPv6) specification. Technical report, 2017.

[13] A. Dunkels. The ContikiMAC Radio Duty Cycling Protocol. page 11.

[14] A. Dunkels, B. Gronvall, and T. Voigt. Contiki - a lightweight and flexible operating system for tiny networked sensors. In *29th Annual IEEE International Conference on Local Computer Networks*, pages 455–462, Tampa, FL, USA, 2004. IEEE (Comput. Soc.).

[15] A. Durand, P. Fasano, I. Guardini, and D. Lento. IPv6 Tunnel Broker. Technical Report RFC3053, RFC Editor, Jan. 2001.

[16] F. Fainelli. The OpenWrt embedded development framework. In *Proceedings of the Free and Open Source Software Developers European Meeting*, 2008.

[17] P. K. Kamma, C. R. Palla, U. R. Nelakuditi, and R. S. Yarrabothu. Design and implementation of 6lowpan border router. In *2016 Thirteenth International Conference on Wireless and Optical Communications Networks (WOCN)*, pages 1–5, Hyderabad, Telangana State, India, July 2016. IEEE.

[18] M. Kovatsch. Demo abstract: Human-CoAP interaction with Copper. In *2011 International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS)*, pages 1–2, June 2011.

[19] E. Nordmark and R. Gilligan. Basic Transition Mechanisms for IPv6 Hosts and Routers. Technical Report RFC4213, RFC Editor, Oct. 2005.

[20] S. Raza, H. Shafagh, K. Hewage, R. Hummen, and T. Voigt. Lithe: Lightweight Secure CoAP for the Internet of Things. *IEEE Sensors Journal*, 13, Oct. 2013.

[21] O. Troan and R. Droms. IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6. Technical Report RFC3633, RFC Editor, Dec. 2003.

[22] Z. Yang and C. H. Chang. 6lowpan Overview and Implementations. 2019. To be published.

[23] Z. Yang, X. Guo, X. Guo, D. Janowsky, and C. H. Chang. A Web Platform for Globally Interconnected 6lowpans. 2019. To be published.