Connected Smart Home over Matter Protocol

Wondimu Zegeye, Ahamed Jemal, Kevin Kornegay
Department of Electrical and Computer Engineering)
Morgan State University
Baltimore,MD, USA
{wozeg1,ahjem1, kevin.kornegay}@morgan.edu

***Abstract*—Project Connected Home over IP, known as Matter, a unifying standard for the smart home, will begin formal device certification in late 2022. The standard will prioritize connectivity using short-range wireless communication protocols such as Wi-Fi, Thread, and Ethernet. The standard will also include emerging technologies such as Blockchain for device certification and security. In this paper, we rely on the Matter protocol to solve the long-standing heterogeneity problem in smart homes. This work presents a hardware Testbed built using development kits, as there is currently very few devices supporting Matter protocol. In addition, it presents a network architecture that automates smart homes to cloud services. The work is a simple and cheap way of developing a Testbed for automating smart homes that uses Matter protocol. The architecture lays the foundation for exploring security and privacy issues, data collection analysis, and data provenance in a smart home ecosystem built on Matter protocol.**

***Index Terms*—Smart Home, IoT, Matter Protocol, Connected Home, IPv6.**

I. Introduction

In the United States, smart home devices are becoming more common. It has recently gained popularity. However, it continues to struggle due to the need for a robust interoperability ecosystem. The ecosystem is fragmented, which presents difficulties for inter-device communications from different vendors. A recent survey poll by Android Authority finds that 87% of its readers own one or more smart home products [1]. Only 5% own one product, and 71% indicate owning five or more products. Similarly, Interpret's Smart Home Matrix™ consumer study reveals that over half of all US consumers, ages 18 to 65, report owning one or more smart home products [2].

Consumers' interest in technology, the desire to safeguard their homes, to protect their families, and the ability to control home systems remotely are the primary motivators for purchasing Smart Home gadgets. These gadgets include but not limited to entertainment devices such as smart speakers and smart TVs, voice-controlled assistants, security cameras, sensors, smart fridges, smart power meters.

While the number of brands and network protocols in connected homes is fantastic for customer choice, it has resulted in a fragmented industry. Interoperability is about more than just connecting devices. The lack of cross-product capabilities combined with a really customer-centric whole-home App experience jeopardizes the customer's aspirations to create a truly smart home.

Interfacing programs such as the if-this-then-that (IFTTT) and Comcast's Stringfy, to mention a few, have typically been used by vendors to address interoperability. Much of the industry has focused on resolving challenges with connected home device compatibility due to disparities in communication protocols (Wi-Fi, Bluetooth, BLE, Zigbee, Z-Wave).

Some works tried to address the interoperability problem by developing frameworks that identify selected characteristics of the various communication protocols in the smart home [3]. Other approaches use App development to control smart home devices [4].

Modern-day homes have smart things, like a thermostat, security cameras, wireless sensors, and switches. Unfortunately, many of these smart things come with different mobile applications, so we have to switch between them to view the whole picture. Imagine having a virtual version of your home that allows the monitoring and controlling of different devices inside it remotely. Did you forget to turn off the heater or air conditioning? Did you forget to close the taps? Wonder how long the kids watched TV?

The research work presented in this paper addresses several critical points for the reader. Firstly, it introduces the Matter protocol for smart home interoperability and heterogeneity problems. Secondly, it demonstrates a very cheap way of developing a Testbed that represents a smart home ecosystem. The results show how we managed to control the Matter protocol enabled smart door lock application. Silicon Laboratories' [5] contribution inspired the development of this Testbed. Lastly, it presents a network architecture representing a smart home's digital twin. Furthermore, the Matter protocol includes Blockchain for Distributed Hyperledger. The network architecture presented addresses this technology.

The organization of this paper is as follows: Section II describes the connected home over IP or Matter protocol and presents the Matter Testbed developed using Silicon Labs development kits. Section III presents traditional smart home devices' automation and the architecture for automating smart home over Matter protocol. Finally, Section IV shows connectivity tests between smart home devices and operation on smart home devices running Matter-based applications such as a smart door lock and a light bulb.

II. Connected Home Over IP

A. Matter Protocol

Matter, previously known as Project Connected Home over IP (CHIP), is a communication standard to improve interoperability between different classes of IoT devices [6].

The Connectivity Standard Alliance (CSA), oversees the project, aims to simplify the IoT industry by establishing uniform standards for manufacturers and consumers.

Manufacturers who follow the standard will be able to connect their Matter devices with devices from other manufacturers, enhancing the usefulness and speed of their IoT infrastructure.

Consumers will not need to use different interfaces to connect with their smart homes if their equipment follow the same standard. Rather than having apps for each manufacturer they buy from, customers can use a single app to interface with all Matter devices.

To get the most out of both high bandwidth and low latency connections, Matter will use Wi-Fi and Thread networks. With such promise, it was no surprise that Apple, Amazon, Google, and Samsung partnered with the CSA to create Matter.

Matter protocol is the new IoT standard, and will improve the interoperability, security, and ease of use across smart home devices, irrespective of the manufacturer of the device and the underlying technology it uses. Matter as an application layer uses Thread protocol as a networking layer.

Based on current IETF standards, Thread provides a mesh networking protocol with the capacity to interact using IPv6 within the Thread network and IPv4/IPv6 with devices outside the network. Advanced encryption, human authentication, many redundancy options, high scalability, low energy consumption, wireless device-to-device, and device-to-cloud communication, and other features are all provided by Thread.

Thread, developed upon the 6LowPAN network, uses specific IPv6 global addresses for devices in the network [7]. It eliminates the need for Network Address Translation (NAT) and gives users and devices direct access to the devices. An IPv6-compliant network is necessary for this. The worldwide IPv6 and IPv4 addresses on the network router boost its compatibility with other networks connected to the internet. NAT transitions will enable IPv6 communication inside the Thread network and IPv4 communication outside when encountering IPv4 communication networks.

The Matter development project is different from other efforts to create standards. Included in the efforts are the Matter standard itself with app functionality and a device library. The GitHub open-source software project enables anyone to use Matter before devices may use the Matter brand on their products. A testing and certification effort must be made that includes a test harness to validate and certify them [8].

Since Matter relies on the IPv6 standard, it is independent of the underlying communication medium. It supports application-to-application communication, whether one application transmits data over Wi-Fi and the other gets it through Thread. Support for Wi-Fi, Ethernet, and Thread will be offered initially (a low-power, low data rate network built on IEEE 802.15.4 wireless devices). As Matter expands, users can still utilize ZigBee and other deployed devices since Matter also offers standard bridging technology that makes them seem like native Matter devices.

B. Experimental Testbed - Smart Home over Matter

The experimental Testbed in Figure 1 shows the deployed connected home over Matter protocol. It builds Matter Lock-app projects (a smart door lock that runs a Matter protocol application) using Silicon Labs EFR32 SLWRB4170A. The deployment follows Silicon labs tools and deployment configurations. To set up the environment, the following software and hardware are required:

1) **Software Requirements**: Simplicity Studio 5, built on Eclipse and C/C++ Development Tooling (CDT), simplifies developer workflow. This software runs on a Windows 10 machine and provides different SDKs to the developer, such as the Gecko SDK. On the Window's host machine, VirtualBox runs

an Ubuntu 20.04 LTS, and Gecko SDK for Linux is also available. Other Software for logging and viewing during the device configuration are Tera Term and J-Link RTT Viewer. SSH clients, such as Putty, can remotely access the nodes in the smart home network.

2) **Hardware Requirements**: SLWSTK6000B Wireless Starter Kit main board is the development kit that allows us to mount Gecko boards such as BRD4170A that can be configured as Matter-enabled devices or as Radio Co-processor (RCP) devices. For example, an RCP device can couple with the Raspberry Pi 4 device to set up Open Thread Border Router. The cables needed to connect the routers and the development kits during the configuration are Micro-USB and USB-Type A cables.

The Windows 10 machine runs the Simplicity commander software for WSK firmware update, create Thread projects, debug nodes, and console to connected boards. The Linux VM also runs on this machine via VirtualBox. The Matter application project is set up on the VM by building the current open source project[8]. Once the build preparation is complete, applications such as EFR32 Lock-App are created.

The preparation of the Thread Border Router requires two pieces of hardware, a recommended Raspberry Pi 3 Model B+ or above) and a Thread-capable Silicon Labs Radio Co-processor (RCP) device such as SLWSTK6000B. On the Raspbian operating system of the Raspberry Pi, the open-source implementation of the Thread networking protocol is installed.

The OTBR requires a radio interface that is achieved via the configuration of one of the wireless development kits as an RCP using the Simplicity Studio. The connection between the RCP and Raspberry Pi is a direct USB cable.

The OpenThread border router also features a Thread Border Agent that supports external commissioning. The Commissioner application is embedded in the Raspberry Pi, contrary to the external Commissioner, a web or mobile application. An OpenThread Border Router (OTBR) features a Thread Border Agent, which supports external Thread Commissioning. In external Thread Commissioning, a device outside the Thread network (for example, a mobile phone) commissions new devices to join the network.

The Thread Commissioner serves to authenticate a user (external Commissioner) or a Thread device onto the network. After authentication, the Commissioner instructs the Border Router to directly transfer Thread network credentials, such as the network key, to the device. This is an example of in-band commissioning, where Thread network credentials are transferred between devices over the radio.

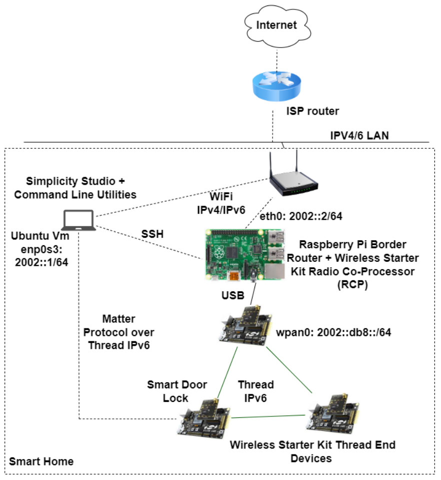

Internet
ISP router
IPV4/6 LAN
Simplicity Studio + Command Line Utilities
WiFi IPv4/IPv6
eth0: 2002::2/64
Ubuntu Vm enp0s3: 2002::1/64
SSH
Raspberry Pi Border Router + Wireless Starter Kit Radio Co-Processor (RCP)
USB
Matter Protocol over Thread IPv6
wpan0: 2002::db8::/64
Smart Door Lock
Thread IPv6
Wireless Starter Kit Thread End Devices
Smart Home

Fig. 1. Smart Home Testbed over Matter protocol.

III. Automating Smart Home Automation Network Architecture

This solution uses a low-cost Siemen's LOGO! Controller that supports industrial control system (ICS) protocols such as Simatic S7 and Modbus TCP/IP. It also supports IoT protocols such as MQTT. The controller is the orchestrator of the IoT devices in the smart home automation and links to the AWS IoT core interface.

The LOGO! controller controls home automation equipment and ingests data into AWS IoT Core. AWS IoT Core collects data at scale and routes messages to multiple AWS services. Services such as AWS Lambda are called inside the AWS IoT Core statement to transform the incoming data before ingestion. While Amazon Timestream stores time series data and optimizes it for fast analytical queries, AWS IoT SiteWise models and stores data from equipment for large-scale deployments. Grafana, installed on Amazon Elastic Compute Cloud (Amazon EC2), visualizes data in near real-time using interactive dashboards. The Alexa Skills Kit (ASK) allows interaction with devices using voice commands.

Different stages of the reference architecture in Figure 3 are as follows:

After data collection and optimization, visualization is an essential component of this architecture. Grafana, installed on Amazon Elastic Compute Cloud (Amazon EC2), visualizes data in near real-time using interactive dashboards. Voice-activated interaction is achieved via Alexa Skills Kit (ASK).

A. Matter Smart Home Automation Network Architecture

The network architecture proposed and partially implemented on a Testbed in this paper presents two network segments. The first segment representing the smart home over Matter protocol is described in Figure 1.

Matter protocol enables IoT devices of the smart home in Figure 1 to connect to the cloud architecture in Figure 3 via HTTPS. The components of this solution depend on Amazon Web Services (AWS). However, equivalent services from other cloud providers such as Microsoft, Google, or Oracle could replace this architecture. Hence, the cloud architecture is vendor agnostic.

AWS API Gateway is a fully managed service for developers to create, publish, maintain, monitor, and secure APIs at any scale. It manages APIs from different applications of the smart home. Regarding server management, AWS Lambda runs code without thinking about servers or clusters, and AWS Fargate (Serverless) compute engine for building applications without managing servers.

Smart home owners can rent Infrastructure-as-a-service (IaaS) to run AWS EKS Anywhere (Kubernetes) to create and operate Kubernetes clusters on the customer-managed infrastructure. Access to the management of the EKS and other services is provided via AWS Cognito, which offers simple and secure user sign-up, sign-in, and access control. On the other hand, AWS Active Directory enables directory-aware workloads.

It is also necessary to include data storage services for streaming data from smart home devices. AWS S3 object storage stores and retrieves any amount of data from anywhere. Using AWS Glue serverless data integration service, the infrastructure architecture makes it easy to discover, prepare, and combine data for analytic, machine learning, and application development. AWS Athena allows the smart home user to perform interactive queries in the web-based cloud storage service.

To run high-performance applications at any scale, AWS DynamoDB, a fully managed, serverless, key-value NoSQL database, is the best choice. On the other hand, Quantum Ledger DB(QLDB), a fully managed ledger database that provides a transparent, immutable, and cryptographically verifiable transaction log, is selected for the blockchain data component of the architecture.

Additional insights can be extracted from the collected and stored data, such as search and analytics using AWS ElasticSearch. Intelligent models can then use the collected data by running different machine learning algorithms from AWS ML. Moreover, Grafana, a near-real-time interactive visualization tool, can display these insights.

The essential steps of operations in the referenced architecture in Figure 3 are as follows:

a) **Data Ingestion**: data from devices that support programmatic APIs, or Software Development Kits (SDKs) stored in
b) **Data Storage**: data from IoT devices are stored in AWS S3 and AWS DynamoDB for additional analysis.
c) **Data Visualization**: data visualization and monitoring are provided via Grafana.
d) **Machine Learning**: perform machine learning inference using models created, trained, and optimized on the cloud. Pre-trained models stored in AWS S3 also perform inference on collected data.
e) **Log Monitoring**: AWS CloudWatch Logs allows monitoring and troubleshooting the systems and applications using the existing system, application and custom log files.
f) **Blockchain**: QLDB tracks each application data change and maintains a complete and verifiable history of changes over time.
g) **Access and Management Services**: IoT management console with CLI provides access to devices and users.

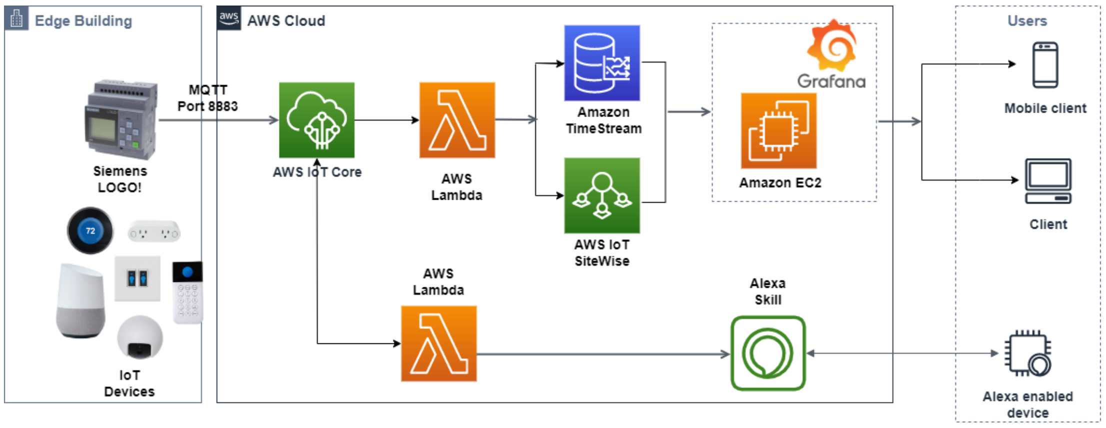


Fig. 2. Smart Home Automation using ICS controller [9].

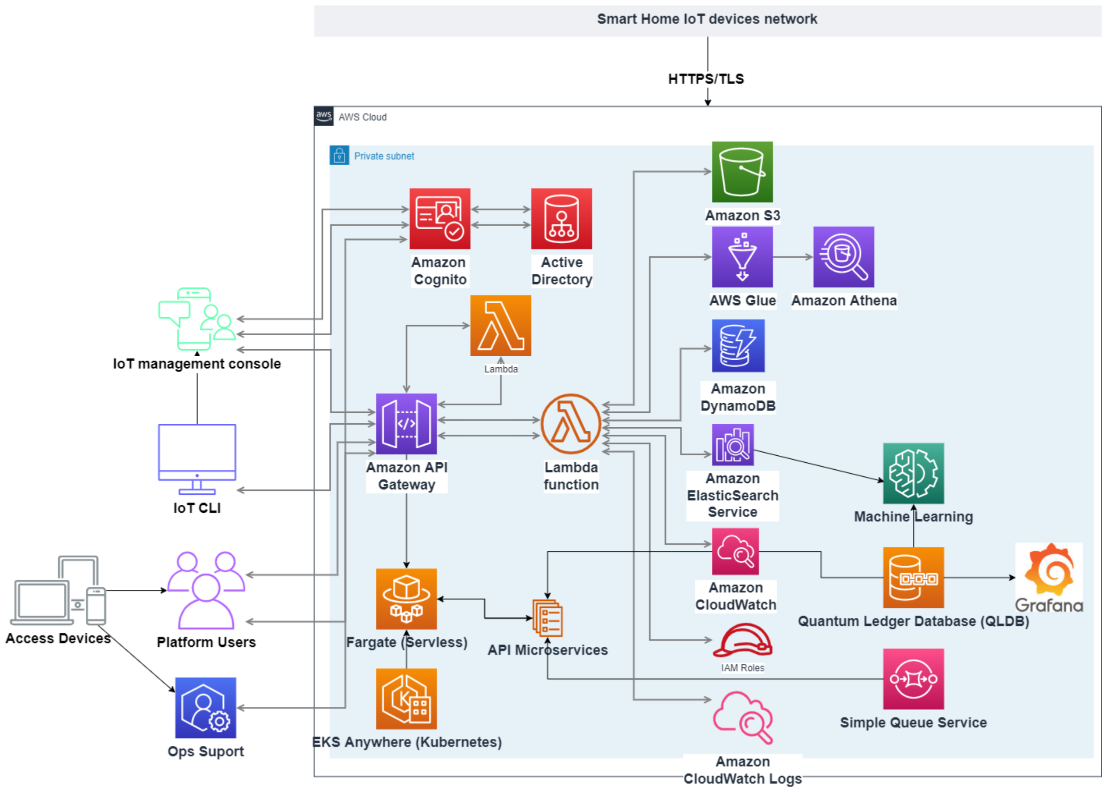


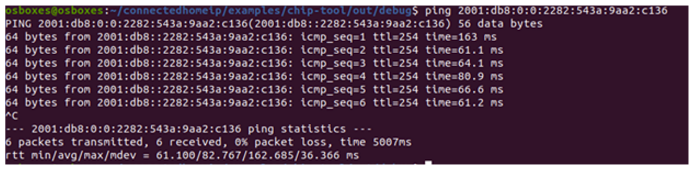
Fig. 3. Smart Home Automation over Matter protocol

IV. RESULTS

As can be seen from Figure 1, the OpenThread mesh network represents a Smart Home with the nodes running Smart Home appliances' applications such as door locks and smart light bulbs. Ubuntu 20.04 guest OS, equipped with Project Connected Home over IP (CHIP) tools, runs on Windows-10 guest via VirtualBox. A bridged connection configuration on the Virtualbox provides access to Wi-Fi to the guest OS.

To access the the smart home's ERF32 Matter door lock App, route interface (epn0s3) is added on the Ubuntu VM with IPv6 address 2002::/1/64. The door lock IPv6 address 2001:db8:00::/64 is added to be reached via network interface 2002::2/64, and its Thread IPv6, fd5c:5a86:64eb::/64, is also added to be reached via network interface 2002::2/64. The ping command snapshot in Figure 4 verifies connectivity.

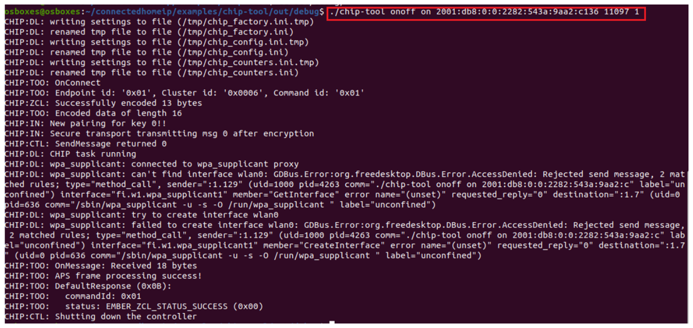
The chip-tool installed on Ubuntu VM allows us to turn the smart door lock ON and OFF. This is achieved



```
osboxes@osboxes:~/connectedhomeip/examples/chip-tool/out/debug$ ping 2001:db8:0:0:2282:543a:9aa2:c136
PING 2001:db8:0:0:2282:543a:9aa2:c136(2001:db8::2282:543a:9aa2:c136) 56 data bytes
64 bytes from 2001:db8::2282:543a:9aa2:c136: icmp_seq=1 ttl=254 time=163 ms
64 bytes from 2001:db8::2282:543a:9aa2:c136: icmp_seq=2 ttl=254 time=61.1 ms
64 bytes from 2001:db8::2282:543a:9aa2:c136: icmp_seq=3 ttl=254 time=64.1 ms
64 bytes from 2001:db8::2282:543a:9aa2:c136: icmp_seq=4 ttl=254 time=80.9 ms
64 bytes from 2001:db8::2282:543a:9aa2:c136: icmp_seq=5 ttl=254 time=66.6 ms
64 bytes from 2001:db8::2282:543a:9aa2:c136: icmp_seq=6 ttl=254 time=61.2 ms
^C
--- 2001:db8:0:0:2282:543a:9aa2:c136 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5007ms
rtt min/avg/max/mdev = 61.100/82.767/162.685/36.366 ms
```

Fig. 4. Connectivity Test

by running the command on/off command from the chip-tool and specifying the IPv6 address of the door lock, 2001:db8:0:0:2282:543a:9aa2:c136, as in Figure 5. Figure 6 shows the capture of the J-Link RTT Viewer during the on/off operation.



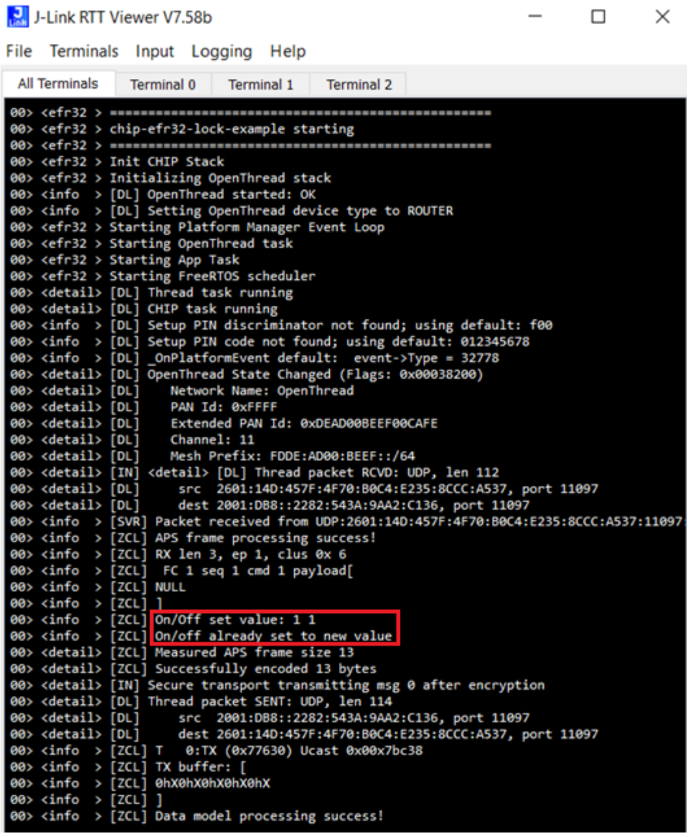
```
osboxes@osboxes:~/connectedhomeip/examples/chip-tool/out/debug$ ./chip-tool onoff on 2001:db8:0:0:2282:543a:9aa2:c136 11097 1
CHIP:DL: writing settings to file (/tmp/chip_factory.ini.tmp)
CHIP:DL: renamed tmp file to file (/tmp/chip_factory.ini)
CHIP:DL: writing settings to file (/tmp/chip_config.ini.tmp)
CHIP:DL: renamed tmp file to file (/tmp/chip_config.ini)
CHIP:DL: writing settings to file (/tmp/chip_counters.ini.tmp)
CHIP:DL: renamed tmp file to file (/tmp/chip_counters.ini)
CHIP:TOO: OnConnect
CHIP:TOO: Endpoint id: '0x01', Cluster id: '0x0006', Command id: '0x01'
CHIP:ZCL: Successfully encoded 13 bytes
CHIP:TOO: Encoded data of length 16
CHIP:IN: New pairing for key 0!!
CHIP:IN: Secure transport transmitting msg 0 after encryption
CHIP:CTL: SendMessage returned 0
CHIP:DL: CHIP task running
CHIP:DL: wpa_supplicant: connected to wpa_supplicant proxy
CHIP:DL: wpa_supplicant: can't find interface wlan0: GDBus.Error:org.freedesktop.DBus.Error.AccessDenied: Rejected send message, 2 mat
ched rules; type="method_call", sender=":1.129" (uid=1000 pid=4263 comm="./chip-tool onoff on 2001:db8:0:0:2282:543a:9aa2:c" label="un
confined") interface="fi.w1.wpa_supplicant1" member="GetInterface" error name="(unset)" requested_reply="0" destination=":1.7" (uid=0
pid=636 comm="/sbin/wpa_supplicant -u -s -O /run/wpa_supplicant " label="unconfined")
CHIP:DL: wpa_supplicant: try to create interface wlan0
CHIP:DL: wpa_supplicant: failed to create interface wlan0: GDBus.Error:org.freedesktop.DBus.Error.AccessDenied: Rejected send message,
 2 matched rules; type="method_call", sender=":1.129" (uid=1000 pid=4263 comm="./chip-tool onoff on 2001:db8:0:0:2282:543a:9aa2:c" lab
el="unconfined") interface="fi.w1.wpa_supplicant1" member="CreateInterface" error name="(unset)" requested_reply="0" destination=":1.7
" (uid=0 pid=636 comm="/sbin/wpa_supplicant -u -s -O /run/wpa_supplicant " label="unconfined")
CHIP:TOO: OnMessage: Received 18 bytes
CHIP:TOO: APS frame processing success!
CHIP:TOO: DefaultResponse (0x0B):
CHIP:TOO:   commandId: 0x01
CHIP:TOO:   status: EMBER_ZCL_STATUS_SUCCESS (0x00)
CHIP:CTL: Shutting down the controller
```

Fig. 5. Smart Door Look On/Off Operation Ubuntu VM

It should be noted that the ping tests verify basic connectivity among the different nodes (Thread/Matter enabled devices in the mesh network. However, Figure 6 shows an actual operation of a Matter Protocol application enabled Smart Door lock. Similarly, we managed to add another node (Matter protocol enabled Smart Light bulb) and operate turning ON/OFF.

Additionally, we prepared a Thread protocol sniffer using Nordic Semiconductor's nRF52840-Dongle to capture the network traffic of the mesh network using Wireshark.

Finally, demonstrating key parameter indicators such as interoperability is different from the goal of this paper as vendor agnostic is, from the beginning, the purpose of the Matter protocol. However, the ping tests among the mesh nodes indirectly demonstrate the inter-device communication feature of the Matter protocol. This feature will allow devices from different vendors to communicate via a single App, such as the VM controller tool.



```
00> <efr32 > ==================================================
00> <efr32 > chip-efr32-lock-example starting
00> <efr32 > ==================================================
00> <efr32 > Init CHIP Stack
00> <efr32 > Initializing OpenThread stack
00> <info  > [DL] OpenThread started: OK
00> <info  > [DL] Setting OpenThread device type to ROUTER
00> <efr32 > Starting Platform Manager Event Loop
00> <efr32 > Starting OpenThread task
00> <efr32 > Starting App Task
00> <efr32 > Starting FreeRTOS scheduler
00> <detail> [DL] Thread task running
00> <detail> [DL] CHIP task running
00> <info  > [DL] Setup PIN discriminator not found; using default: f00
00> <info  > [DL] Setup PIN code not found; using default: 012345678
00> <info  > [DL] _OnPlatformEvent default:  event->Type = 32778
00> <detail> [DL] OpenThread State Changed (Flags: 0x00038200)
00> <detail> [DL]    Network Name: OpenThread
00> <detail> [DL]    PAN Id: 0xFFFF
00> <detail> [DL]    Extended PAN Id: 0xDEAD00BEEF00CAFE
00> <detail> [DL]    Channel: 11
00> <detail> [DL]    Mesh Prefix: FDDE:AD00:BEEF::/64
00> <detail> [IN] <detail> [DL] Thread packet RCVD: UDP, len 112
00> <detail> [DL]     src  2601:14D:457F:4F70:B0C4:E235:8CCC:A537, port 11097
00> <detail> [DL]     dest 2001:DB8::2282:543A:9AA2:C136, port 11097
00> <info  > [SVR] Packet received from UDP:2601:14D:457F:4F70:B0C4:E235:8CCC:A537:11097
00> <info  > [ZCL] APS frame processing success!
00> <info  > [ZCL] RX len 3, ep 1, clus 0x 6
00> <info  > [ZCL]  FC 1 seq 1 cmd 1 payload[
00> <info  > [ZCL] NULL
00> <info  > [ZCL] ]
00> <info  > [ZCL] On/Off set value: 1 1
00> <info  > [ZCL] On/off already set to new value
00> <detail> [ZCL] Measured APS frame size 13
00> <detail> [ZCL] Successfully encoded 13 bytes
00> <detail> [IN] Secure transport transmitting msg 0 after encryption
00> <detail> [DL] Thread packet SENT: UDP, len 114
00> <detail> [DL]     src  2001:DB8::2282:543A:9AA2:C136, port 11097
00> <detail> [DL]     dest 2601:14D:457F:4F70:B0C4:E235:8CCC:A537, port 11097
00> <info  > [ZCL] T   0:TX (0x77630) Ucast 0x00x7bc38
00> <info  > [ZCL] TX buffer: [
00> <info  > [ZCL] 0hX0hX0hX0hX0hX
00> <info  > [ZCL] ]
00> <info  > [ZCL] Data model processing success!
```

Fig. 6. Smart Door Look On/Off Operation J-Link RTT Viewer

V. Conclusions

In Summary, the test cases presented in this section show the successful deployment of a Matter protocol Testbed using development kits and open-source software tools, libraries, and applications. In particular, a Matter protocol smart door lock application is deployed as a smart home device. The test successful uses a software tool to turn ON/OFF the smart door lock and logs these events' results using the J-Link RTT viewer.

This paper presents a cheap Testbed implementation of the Matter protocol based smart home system. It also develops network architecture that automates smart homes via different cloud services. The smart home architecture over Matter tested is already developed. Cloud automation aims to create a digital twin of a smart home is defined. The architecture lays the foundation for security and privacy, data analysis, artificial intelligence, and Blockchain solution over an automated smart home.

References

[1] A. Sharma, We asked, you told us: Here's how many smart home devices you own (androidauthority.com) Catching the Sun, American Society of Mechanical

Engineers, Aug. 2019. Accessed on: Dec. 02, 2021.[Online].Available:https://www.androidauthority.com/smart-home-devices-poll-results-2737270/

[2] Interpret LLC., Smart Home Matrix, Accessed on: Dec. 02, 2021.[Online].Available:https://interpret.la/products/smart-home-matrix/

[3] M. O. Farooq, I. Wheelock and D. Pesch, ”IoT-Connect: An Interoperability Framework for Smart Home Communication Protocols,” in IEEE Consumer Electronics Magazine, vol.9, no.1, pp.22-29, 1 Jan. 2020, doi: 10.1109/MCE.2019.2941393.

[4] J. Kim, S. Y. Ko, S. Son and D. Han, ”Lumos: Improving Smart Home IoT Visibility and Interoperability Through Analyzing Mobile Apps,” 2020 IEEE 28th International Conference on Network Protocols (ICNP), 2020, pp. 1-13, doi: 10.1109/ICNP49622.2020.9259352.

[5] CHP-101:Connect Home over IP (CHIP) Lab, Accessed on: June. 18,2022.[Online].Available:https://www.silabs.com/support/training/connected-home-over-ip-intro/connected-home-over-ip-lab

[6] Matter, The foundation for connected things, Accessed on: Dec. 11, 2021.[Online].Available:https://csa-iot.org/all-solutions/Matter/

[7] “IETF, 6LowPAN,” Accessed on: Aug. 12, 2022. [Online].Available: https://tools.ietf.org/html/rfc8138

[8] GitHub-project-chip/connectedhomeip:Matter (formerly Project CHIP), Accessed on: Dec. 02, 2021. [Online].Available: https://github.com/project-chip/connectedhomeip

[9] Viktoria Semaan, Emad Mankbadi, Automating Your Home with Grafana and Siemens Controllers, Aug. 2021. Accessed on: Dec 11, 2021.[Online]. Available: https://aws.amazon.com/blogs/architecture/automating-your-home-with-grafana-and-siemens-co