Secure IoT Firmware For RISC-V Processors

Cesare Garlati
Hex Five Security
Redwood City, CA, USA
cesare.garlati@hex-five.com

Sandro Pinto
Universidade do Minho
Guimarães, Portugal
sandro.pinto@dei.uminho.pt

***Abstract* — Building secure RISC-V devices is challenging as the RISC-V ISA doesn't specify the hardware blocks necessary for the trusted execution of the many 3rd party components of the software stack. RISC-V developers are left alone figuring out how to shield trusted code from unverified 3rd party software libraries. In this paper, we introduce the industry-first secure IoT stack for RISC-V. We describe and explain all hardware and software components necessary to build state-of-the-art device, firmware, and cloud management service. These include RISC-V 32-bit SoC, MultiZone Trusted Execution Environment, TCP/IP connectivity, TLS/ECC cryptography, and MQTT client and broker providing telemetry and OTA applications deployment and firmware updates. All components are built on free and open standards, distributed under permissive licensing, and freely available for download from GitHub.**

Keywords* — *RISC-V, IoT, firmware, security, secure boot, firmware updates, cloud, trusted execution environment, TEE, TCP/IP, TLS, MQTT, OTA, lwIP, mbedTLS, FreeRTOS, open source.

I. Introduction

Building secure IoT firmware for embedded devices is challenging. These resource-constrained devices typically lack the hardware resources necessary for the trusted execution of the many complex 3rd party software components required for secure operations. And state-of-the-art security features like secure boot, remote attestation, authenticated access to commercial cloud services, and over-the-air (OTA) firmware updates require a number of complex 3rd party software components [1,2]. These libraries are difficult to integrate, expose the system to increased attack surface, and inevitably lead to the dangerous execution of trusted and untrusted code in the same chip [1,3].

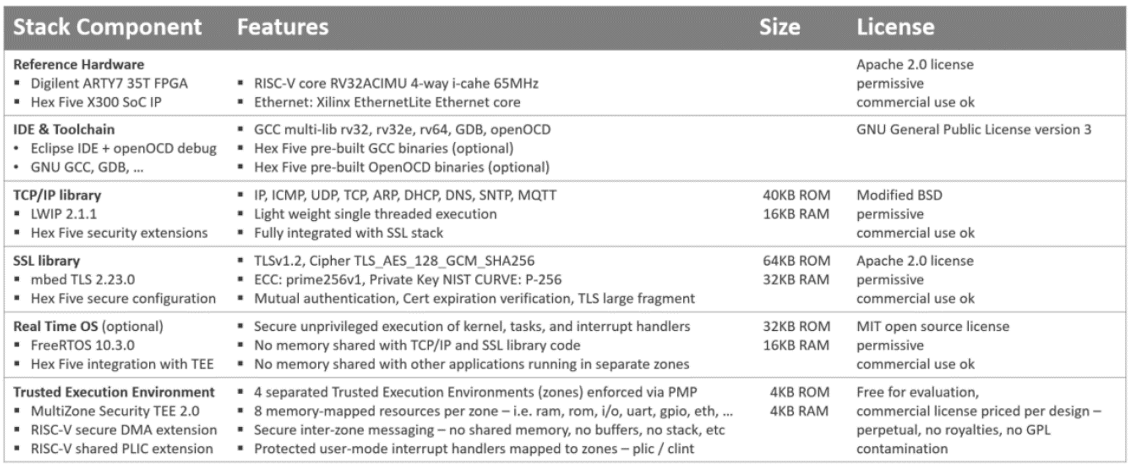
Over time, established platform vendors have developed light-weight Trusted Execution Environments (TEE) and relative embedded software stacks optimized for their smaller processors [4]. However, none of these are available to RISC-V developers who are left alone figuring out how to shield trusted code from unverified 3rd party software libraries and how to safely combine these components into the single firmware image powering their commercial applications.

To meet the high-grade security requirements imposed by new IoT, we propose the MultiZone Secure IoT Firmware for RISC-V. The MultiZone IoT Firmware is the quick and safe way to build secure IoT applications with RISC-V processors. It provides secure access to IoT clouds, real-time monitoring, secure boot, and remote firmware updates. The secure IoT firmware is based on the innovative MultiZone Trusted Execution Environment (TEE) [5] optimized for any 32-bit and 64-bit standard RISC-V processor with U-mode extension. In this paper, we describe all software and hardware components of a reference application that securely controls a robotic arm (optional) via private or public MQTT cloud. The reference application includes the RISC-V open source softcore X300, low-level drivers, the TEE, TCP/IP connectivity, TLS/ECC cryptography, and MQTT client providing real-time monitoring, device management, telemetry, and OTA applications deployment and remote firmware updates. The reference application works with any RISC-V processor and cloud provider, it is built on free and open standards, distributed under permissive licensing, and freely available for download at https://github.com/hex-five/multizone-secure-iot-stack.

II. RISC-V Security Primitives

RISC-V is a modern computer architecture developed at U.C Berkeley and now ratified by the RISC-V foundation [6]. RISC-V differs from other commercial computer architectures in that its instruction set architecture (ISA) is open source. RISC-V specifies a number of security building blocks in the ISA itself. These include (i) *privileged execution levels*, (ii) *physical memory protection* (PMP), and (iii) *user-mode interrupt* delegation.

RISC-V Privilege Levels. At any time, a RISC-V hardware thread (so-called hart), runs at a given privilege level. According to the latest RISC-V Privileged ISA Specification (version 1.12) these include User/Application (U), Supervisor (S), Hypervisor (H), and Machine (M). The Hypervisor (H) is currently under review for ratification. These privilege levels are used to separate and protect the different components of the software



Stack Component	Features	Size	License
Reference Hardware ▪ Digilent ARTY7 35T FPGA ▪ Hex Five X300 SoC IP	▪ RISC-V core RV32ACIMU 4-way i-cahe 65MHz ▪ Ethernet: Xilinx EthernetLite Ethernet core		Apache 2.0 license permissive commercial use ok
IDE & Toolchain • Eclipse IDE + openOCD debug • GNU GCC, GDB, …	▪ GCC multi-lib rv32, rv32e, rv64, GDB, openOCD ▪ Hex Five pre-built GCC binaries (optional) ▪ Hex Five pre-built OpenOCD binaries (optional)		GNU General Public License version 3
TCP/IP library ▪ LWIP 2.1.1 ▪ Hex Five security extensions	▪ IP, ICMP, UDP, TCP, ARP, DHCP, DNS, SNTP, MQTT ▪ Light weight single threaded execution ▪ Fully integrated with SSL stack	40KB ROM 16KB RAM	Modified BSD permissive commercial use ok
SSL library ▪ mbed TLS 2.23.0 ▪ Hex Five secure configuration	▪ TLSv1.2, Cipher TLS_AES_128_GCM_SHA256 ▪ ECC: prime256v1, Private Key NIST CURVE: P-256 ▪ Mutual authentication, Cert expiration verification, TLS large fragment	64KB ROM 32KB RAM	Apache 2.0 license permissive commercial use ok
Real Time OS (optional) ▪ FreeRTOS 10.3.0 ▪ Hex Five integration with TEE	▪ Secure unprivileged execution of kernel, tasks, and interrupt handlers ▪ No memory shared with TCP/IP and SSL library code ▪ No memory shared with other applications running in separate zones	32KB ROM 16KB RAM	MIT open source license permissive commercial use ok
Trusted Execution Environment ▪ MultiZone Security TEE 2.0 ▪ RISC-V secure DMA extension ▪ RISC-V shared PLIC extension	▪ 4 separated Trusted Execution Environments (zones) enforced via PMP ▪ 8 memory-mapped resources per zone – i.e. ram, rom, i/o, uart, gpio, eth, … ▪ Secure inter-zone messaging – no shared memory, no buffers, no stack, etc ▪ Protected user-mode interrupt handlers mapped to zones – plic / clint	4KB ROM 4KB RAM	Free for evaluation, commercial license priced per design – perpetual, no royalties, no GPL contamination

Fig. 2. Multi-zone Secure IoT Firmware for RISC-V: summary of software components, features, size, and license

stack from each other. A synchronous non-maskable exception (trap) is raised if an instruction would result in the hart performing operations not permitted by the current privilege level.

RISC-V Physical Memory Protection (PMP). The PMP filter allows the highest privilege level (M) to protect specific memory regions and to grant lower privilege levels access only to specific contiguous memory regions according to read / write / execute policies. This allows the partition of functionality between execution environments and other functional component behavior.

RISC-V User-Level Interrupts. A third relevant feature is the support for user-level interrupts - "N" extension. This extension allows the interrupt controller to delegate exception handling to more secure user-level handlers, bypassing the highest privilege level in the outer execution environment.

III. MULTI ZONE SECURE IOT FIRMWARE FOR RISC-V

The MultiZone Secure IoT Firmware for RISC-V includes the following software components: the MultiZone TEE, the lwIP TCP/IP stack, the mbedTLS crypto library, the FreeRTOS real-time operating system - optional, MQTT support, and three bare metal applications demonstrating protected access to hardware and software resources. Fig. 1 presents a summary of the main software components of the IoT firmware including security extensions, memory footprint, and license type. The architecture is modular and can be easily configured to add or remove individual components to fit a broad range of devices and applications. The MultiZone IoT Firmware works with any 32-bit and 64-bit RISC-V processor with standard U-mode extension. For a quick start, we recommend the development kit based on the free and open source softcore SoC X300. This is an enhanced version of the E300 SoC (Rocket RV32) originally developed at U.C. Berkeley. Like the E300, the X300 is designed to be programmed onto a Xilinx Artix-7 FPGA. The X300 RTL is entirely free for commercial and non-commercial use.

MultiZone TEE. The MultiZone Security Trusted Execution Environment [5] provides hardware-enforced software-defined separation of multiple functional areas within the same chip.

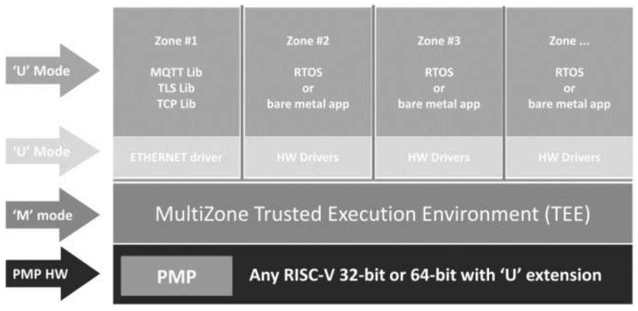


Fig. 1. Multi-zone Security for RISC-V: system architecture

MultiZone is completely self-contained, exposes an extremely small attack surface, and is policy-driven, meaning that no coding or security expertise are required. The main components include: (i) the TEE Configurator, a development utility that extends the GNU toolchain, (ii) the TEE Runtime, a small binary providing secure boot, separation kernel, and secure communications; and (iii) the TEE API, a free and open API providing static wrappers for system calls. The MultiZone IoT Firmware for RISC-V includes the MultiZone Security TEE 2.0 for RISC-V providing: 4 separated Trusted Execution Environments (zones) enforced via PMP; 8 memory-mapped resources per zone to protect programs, data, peripherals, and DMA and interrupt controllers; secure inter-zone messaging with no shared memory structures like buffers, hype, or stack; protected user-mode interrupt service routines mapped to zones [7] - up to 128. Fig. 2 illustrates the overall system architecture of the MultiZone Security TEE for RISC-V.

TCP/IP Library. The lightweight IP (lwIP) library is a permissive open-source implementation of the TCP/IP protocol stack. lwIP is designed to minimize code footprint making it suitable for embedded systems with limited resources. The MultiZone IoT Firmware includes a modified version of lwIP 2.1.1 optimized for security and performance. In particular, lwIP is configured for single-threaded execution, avoiding overhead and security risks typically associated with multi-tasking operating systems like FreeRTOS, which is entirely

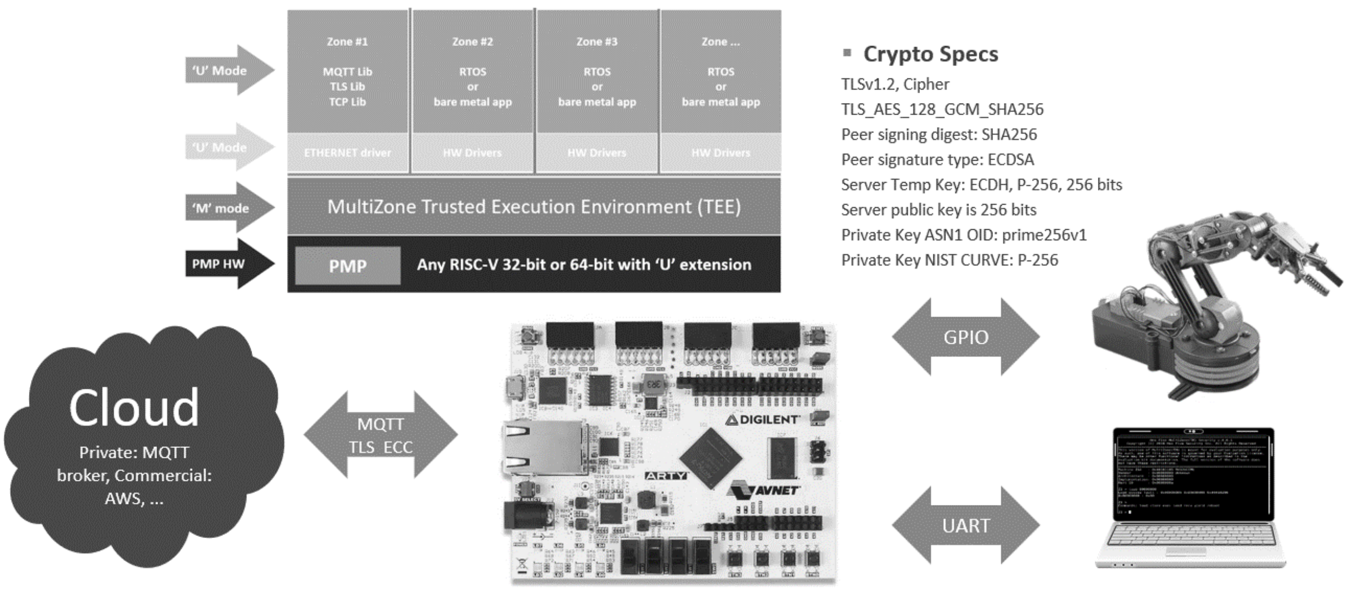

'U' Mode
Zone #1
MQTT Lib
TLS Lib
TCP Lib
Zone #2
RTOS
or
bare metal app
Zone #3
RTOS
or
bare metal app
Zone ...
RTOS
or
bare metal app
'U' Mode
ETHERNET driver
HW Drivers
HW Drivers
HW Drivers
'M' mode
MultiZone Trusted Execution Environment (TEE)
PMP HW
PMP
Any RISC-V 32-bit or 64-bit with 'U' extension
Crypto Specs
TLSv1.2, Cipher
TLS_AES_128_GCM_SHA256
Peer signing digest: SHA256
Peer signature type: ECDSA
Server Temp Key: ECDH, P-256, 256 bits
Server public key is 256 bits
Private Key ASN1 OID: prime256v1
Private Key NIST CURVE: P-256
Cloud
Private: MQTT broker, Commercial: AWS, ...
MQTT
TLS ECC
GPIO
UART
DIGILENT
ARTY
AVNET

Fig. 3. Multi-zone Secure IoT Firmware for RISC-V - Reference Application

optional and included only to support legacy applications. The connectivity stack is pre-configured to provide IP, UDP, TCP, ARP, ICMP, DHCP, DNS, SNTP, and MQTT. Additional services can be enabled if necessary, i.e., IPV6, PPP, and HTTP/HTTPS server. The connectivity stack is fully integrated with the cryptography library to provide core PKI functionality and advanced TSL functionality like mutual authentication and TLS large packet fragmentation.

TLS Library. Mbed TLS is a permissive open-source library that implements cryptographic primitives, X.509 certificate manipulation, and the SSL/TLS and DTLS protocols. The MultiZone IoT Firmware integrates a modified version of mbed TLS 2.23.0. The library is configured to provide TLS 1.2 connectivity optimized for embedded devices. The default cipher suite provides state-of-the-art secure connectivity required to connect to commercial IoT cloud services like AWS, Azure, and similar – i.e. Advanced Encryption Standard with 128bit key in Galois/Counter mode (AES 128 GCM) and Secure Hash Algorithm 256 (SHA256). Default keys are based on the 256-bits NIST curve of the Elliptic Curve Cryptography (ECC) implementation. The standard configuration supports client authentication, server name authentication, certificate expiration verification, TLS large fragments, and high-grade entropy.

MQTT Gateway. The MultiZone IoT Firmware zone #1 implements a single-threaded bare-metal MQTT gateway. The gateway is responsible for forwarding MQTT messages (including binary files) to and from the MQTT broker. The gateway is written in C and it is fully configurable. Internal messages sent to zone #1 are forwarded to the broker topic mapped to the device-id and source zone. External messages sent via the broker to the topic device-id/zone are forwarded internally to the respective zone. Individual applications deployment can be triggered manually via authenticated MQTT posts or automatically upon device startup via MQTT persistent messages. This allows to ship devices with a minimum software image and automatically provision the most up-to-date version of the application software upon initial connection to the broker.

Real-Time Operating System. FreeRTOS is an open-source real-time operating system (RTOS) for microcontrollers and small microprocessors. FreeRTOS includes a multi-tasking kernel and an extensive number of software libraries suitable for use across industry sectors and applications. Thanks to the built-in trap & emulation engine, the MultiZone IoT Firmware can run one or more instances of unmodified FreeRTOS binaries and relative tasks (kernel 10.4.1). Kernel, tasks, and interrupts run in protected unprivileged U-mode. FreeRTOS is included in the firmware to facilitate security upgrades of existing legacy applications. It is not a requirement as the MultiZone TEE provides its on safety-critical preemptive scheduler – se for example the alternative implementation of the 3-task real-time controller of the robotic arm which is provided as a FreeRTOS application (zone 4.2) or MultiZone TEE application (zone 4.1).

IV. USE CASES

The MultiZone Secure IoT Firmware is suitable for a wide range of IoT applications that require high-grade security and compliance to common methods and open standards.

Secure access to commercial IoT clouds. IoT devices generate a vast amount of data. The Cloud, as part of the IoT ecosystem, manages the flow, process, analysis, and storage of these data. Thus, secure access to commercial IoT clouds is critical for several IoT service providers and device makers, in particular those concerned about backdoors and the lack of separation of consolidated 3rd party software. State-of-the-art technology relies on lightweight communication protocols (i.e., MQTT) atop of secure communications layers. MultiZone Secure IoT Firmware provides built-in secure connectivity (TLS, ECC, mutual authentication) compliant to the requirements of commercial cloud provides like AWS and Azure, as well as protected execution of 3rd party components via separated hardware-enforced execution environments.

Remote firmware updates. Firmware updates are one of the most important features to take into account while developing any IoT device. In particular, in some countries, there is already strict legislation that dictates remote firmware updates (OTA) as mandatory. For IoT device makers concerned about the time, cost, and security risk of developing a DIY solution, MultiZone Secure IoT Firmware provides high-grade security OTA updates via open standard MQTT and TLS protocols. The OTA update mechanism is not locked to any commercial cloud service and work with commercial and private IoT cloud.

Real-time monitoring and device management. IoT devices connect to the network to provide information they gather from the environment through sensors, or to allow other systems to reach out and act on the world through actuators. MultiZone Secure IoT Firmware provides secure bidirectional access to/from the device via standard MQTT protocol and enables real-time monitoring, telemetry, and device management. To mitigate the recurring costs of commercial web services, the firmware serves a broader audience by working with public and private clouds, i.e., OEM owned PKI and backend infrastructure.

V. Reference Application and Evaluation

Fig. 3 depicts the MultiZone Secure IoT Firmware reference application on a Digilent ARTY7 35T FPGA running a RISC-V SoC, i.e., the X300. The MultiZone Secure IoT firmware includes all software and hardware components necessary to build a complete IoT application that securely controls a robotic arm via a standard MQTT broker in the cloud. The inexpensive USB robotic arm is completely optional but a lot of fun to build and operate. *Zone 1* connects to a private or commercial IoT cloud via Ethernet or wirelessly if a Wi-Fi router is connected to the board Ethernet port. Operating in zone 1 is the fully integrated, single-threaded, secure network stack providing TCP/IP, TLS, and MQTT. *Zone 2* connects to a local host via UART. Operating in zone 2 is a bare metal ANSI terminal console written in C. It presents the user with a command-line interface to send and receive MQTT messages, to assess the enforcement of the separation policies, the security of protected DMA and shared PLIC controller, and to measure the overall performance overhead. Zone 3 blinks an LED and interfaces three additional LEDs using local buttons to demonstrate real time multi-tasking, protected interrupt handling, and secure messaging. It has two interrupts mapped to button S1 and S2 that toggle LED2 and LED3 and send a notification message to both zone 1 (forwarded to the broker) and zone2. In addition, zone3 implements a few message responders to verify separation policies, non-interference, and preemptive execution. Zone 4 operates the optional robot interfaced through an SPI-to-USB controller. Commands can be submitted to the USB robot remotely via the broker (sub-topic zone 4) or locally via the terminal application in zone 2. The status of the robot is reported back via secure messaging and published to the broker. Multi-tasking for zone 4 is provided by the TEE kernel or by FreeRTOS.

A. Evaluation

The MultiZone Secure IoT Firmware for RISC-V was evaluated on a Digilent ARTY7 35T FPGA, running the X300 SoC. The X300 is equipped with a RISC-V core RV32ACIMU, 4-way i-cache, and it is clocked at 65MHz. In addition to the extensive tests for security, separation, and reliability accessible via zone #2, we measured the MultiZone Secure IoT Firmware memory footprint, context switch overhead, and connectivity round-trip time (RTT).

MultiZone Secure IoT Firmware Size. By design, the MultiZone Secure IoT Firmware for RISC-V is extremely optimized for resource-constrained devices. The default configuration requires approximately 100 KiB of FLASH and 32KiB of RAM, enabling the deployment of connected solutions in devices with few KiB of memory.

Round-Trip Time. To assess the round-trip time, we ran the *ping* command from a remote terminal. For the system under test, configured for four zones, the ping statistics have reported an average RTT of 1.20 milliseconds.

References

[1] Chris Conlon and Cesare Garlati. “A New Zero-Trust Model for Securing Embedded Systems”. In Proceedings of the Embedded World Conference, Nuremberg, Germany, 2019.

[2] M. N. Aman et al., "HAtt: Hybrid Remote Attestation for the Internet of Things With High Availability," in IEEE Internet of Things Journal, vol. 7, no. 8, pp. 7220-7233, Aug. 2020.

[3] O. Alrawi, C. Lever, M. Antonakakis and F. Monrose, "SoK: Security Evaluation of Home-Based IoT Deployments." IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 2019.

[4] Sandro Pinto and Nuno Santos, “Demystifying Arm TrustZone: A Comprehensive Survey.” ACM Computing Surveys, vol. 51, no. 6, article 130, December 2018.

[5] Cesare Garlati and Sandro Pinto. “A Clean Slate Approach to Linux Security RISC-V Enclaves”. In Proceedings of the Embedded World Conference, Nuremberg, Germany, 2020.

[6] Krste Asanović and David A. Patterson. "Instruction sets should be free: The case for RISC-V." EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2014-146, 2014.

[7] Sandro Pinto and Cesare Garlati. “User Mode Interrupts: A Must for Securing Embedded Systems”. In Proceedings of the Embedded World Conference, Nuremberg, Germany, 2019.

[8] GitHub repository containing RTL and bitstream of the RISC-V X300 SoC platform: https://github.com/hex-five/multizone-fpga