周报 向嘉豪(2025-10-13)

摘要: 本周完成 ML-DSA 论文实验方法论体系构建与实验环境部署准备工作。论文方面：完成第 4 节实验方法论框架的系统性设计，建立 ARM Cortex-M4 微控制器性能评估完整体系。实验平台确定采用 STM32F407VG 开发板(168MHz 主频、1MB Flash、192KB SRAM)配合 pqm4 优化库实现三组 ML-DSA 参数集基准测试。性能测量框架利用 DWT 硬件单元实现周期级精度计时，采用栈水印技术与静态分析相结合的双重内存剖析方法。基准测试设计覆盖密钥生成、签名生成、签名验证全流程，包含 10 字节最小传感器读数、50 字节典型遥测数据、100 字节扩展状态报告三类 IoT 载荷场景，建立 ECDSA P-256 基线对比基准。集成测试协议实现 MQTT 载荷嵌入式签名端到端验证流程，评估指标体系涵盖计算性能(CPU 周期、执行时间)、内存占用(代码尺寸、栈消耗)、协议级开销(消息尺寸、传输延迟)三大维度。实验环境方面：启动基于方法论框架的实际硬件平台搭建工作，当前处于 STM32 开发板配置与软件工具链部署阶段，pqm4 库移植与 MQTT 客户端集成正在进行中。

下周计划: 1) 完成 STM32F407VG 开发板实验环境搭建，完成 pqm4 库 ARM Cortex-M4 平台移植与功能验证。 2) 启动 ML-DSA 基准测试实验数据采集，完成密钥生成、签名生成、签名验证三大操作的性能基线测量。 3) 开展 MQTT 协议集成实验，实现载荷嵌入式签名原型系统，测量协议级传输开销。

1 ML-DSA 论文第 4 节实验方法论体系完成

1.1 实验平台架构设计

硬件平台选型与配置 本周完成实验平台核心架构设计，确定采用 ARM Cortex-M4 微控制器作为目标评估平台，该平台代表工业监控、智慧农业、建筑自动化等领域部署的中端 IoT 设备典型配置。具体硬件选型为 STM32F407VG 开发板，搭载 ARM Cortex-M4F 内核运行于 168MHz 主频，配备硬件浮点运算单元、1MB Flash 程序存储与 192KB SRAM 运行时内存，该配置真实反映当代 IoT 部署中需要密码学认证能力的资源约束环境。软件环境采用 ARM GCC 工具链 10.3.1 版本，启用-O3 激进性能优化标志以实现标准兼容前提下的最佳性能表现。

密码学库集成方案 ML-DSA 实现源自 pqm4 参考库，该库提供针对 ARM Cortex-M4 优化的全部三个标准化参数集(ML-DSA-44、ML-DSA-65、ML-DSA-87)实现，并通过 NIST 测试向量完整性验证。为建立公平性能对比基准，集成 micro-ecc 库实现 ECDSA NIST P-256 曲线，采用相同编译器优化设置确保测量可比性。MQTT 协议集成采用 Eclipse Paho MQTT 嵌入式 C 客户端库，配置 QoS 1 服务质量级别与 5 秒保活间隔，通信目标为部署于专用服务器的 Mosquitto MQTT 代理 2.0.15 版本。网络连接利用 ESP32-WROOM-32 无线模块提供 IEEE 802.11n WiFi 接入，消除物理布线约束同时维持真实 IoT 通信模式。

1.2 性能测量框架建立

计算性能精确测量机制 计算性能测量利用 ARM Cortex-M4 数据观察点与跟踪(DWT)硬件单元，通过 DWT_CYCCNT 寄存器提供周期精度执行时间量化。该硬件测量方法消除软件性能剖析相关开销，实现单周期时序分辨率。测量范围涵盖完整操作执行流程，包括全部预处理、核心算法计算、结果格式化阶段，确保端到端性能特征的准确捕获。

内存占用双重剖析方法 内存利用分析采用静态与动态测量技术相结合的双重方法论。静态内存消耗通过 arm-none-eabi-size 工具链实用程序从编译输出中提取，量化 ML-DSA 实现代码与初始化数据段的 Flash 内存需求。动态内存剖析采用栈水印技术，在操作执行前以特征模式(0xDEADBEEF 哨兵值)初始化栈内存区域，执行后扫描模式破坏范围确定峰值栈利用。堆分配监控通过插桩 malloc 与 free 函数跟踪运行时内存请求，尽管 ML-DSA 嵌入式实现通常避免动态分配。能耗评估采用 INA219 电流传感器模块以 100Hz 采样频率测量密码学操作执行全程供电电流，电压监控与电流积分提供操作级能耗估算，支持电池供电 IoT 设备的功率约束部署分析。

1.3 基准测试方案设计

密码学操作全流程评估 基准测试方法论涵盖全部 ML-DSA 密码学操作在标准化参数集上的系统性评估。密钥生成基准测试测量完整公私钥对生成流程，包括随机种子生成、矩阵扩展、多项式采样操作。签名生成基准测试捕获端到端签名延迟，涵盖消息散列、拒绝采样迭代、签名编码阶段，统计分析计入可变迭代次数影响。验证基准测试测量签名验证计算成本，包括签名解码、多项式重构、有效性检查操作。

每个参数集(ML-DSA-44、ML-DSA-65、ML-DSA-87)横跨代表性 IoT 消息载荷接受评估，涵盖 10 字节最小传感器读数(单一温度值)、50 字节典型遥测包(多传感器聚合数据)、100 字节扩展状态报告(设备诊断与元数据)。该载荷多样性捕获真实 IoT 通信模式，同时支持跨消息尺寸的开销比率分析。基线性能对比采用 ECDSA P-256 实现执行相同操作序列，密钥对生成产生 32 字节私钥与 64 字节公钥，签名生成处理等效消息长度产生 64 字节 DER 编码签名，验证操作处理相同签名数据。全部测量在相同环境条件下执行，保持一致时钟配置与编译器优化设置。

统计严谨性保证 统计严谨性源自重复测量方法论，每项操作执行 1000 次迭代并消除离群值。报告中位数执行时间提供对测量噪声具有韧性的稳健中心趋势估计，配合四分位距量化性能变异性。对于因拒绝采样表现显著方差的签名生成操作，额外报告观察到的最小与最大执行时间以刻画性能边界。

1.4 MQTT 集成测试协议确立

端到端认证工作流实现 MQTT 协议集成测试评估在 MQTT 消息载荷内嵌入 ML-DSA 签名的端到端认证工作流。签名集成架构实现载荷嵌入方法以维持 MQTT 协议兼容性：发布者生成消息内容的 ML-DSA 签名，将签名附加到载荷数据，通过标准 MQTT PUBLISH 操作传输复合消息。订阅者接收复合消息，提取嵌入签名，通过预定义密钥分发机制检索发布者公钥，在处理消息内容前验证签名真实性。

端到端延迟测量涵盖完整消息生命周期：发布者设备上的签名生成、MQTT 消息序列化与传输、网络传播延迟、代理处理与路由、订阅者接收与反序列化、签名验证。测量插桩在工作流各阶段捕获时间戳，支持延迟分解与瓶颈识别。协议开销量化对比签名消息传输特征与基线未签名 MQTT 通信。消息尺寸开销计算测量包含签名数据与任何必需元数据(密钥标识符、算法参数、编码格式)的总载荷扩展。吞吐量分析测量连续操作下可持续消息发布速率，识别限制系统可扩展性的计算瓶颈。网络带宽消耗评估量化容量受限 IoT 网络的传输成本影响。

测试场景评估全部三个 MQTT QoS 级别：QoS 0 即发即弃传输测量最小开销场景，QoS 1 确认交付评估签名验证延迟与协议确认超时的交互，QoS 2 恰好一次交付量化对多阶段消息交换协议的计算影响。

1.5 评估指标体系构建

多维性能评估框架 评估框架采用跨越计算性能、内存占用、协议级影响的综合指标套件。计算指标包括 CPU 周期计数提供架构无关性能刻画，执行时间测量(毫秒)反映应用体验的实际延迟，每秒操作数量化连续工作负载下可持续密码学操作速率。内存指标涵盖代码尺寸(千字节)量化 ML-DSA 实现的 Flash 内存需求，常量数据结构与全局变量的静态 RAM 分配，操作执行期间栈内存峰值消耗，合并全部存储需求的总内存占用。这些指标支持针对特定微控制器配置约束内存资源的部署可行性评估。

协议级指标刻画 MQTT 集成影响，通过消息尺寸开销(相对未签名基线消息的百分比增长)、传输延迟(发布调用到订阅者接收时间)、验证延迟(消息接收到验证内容可用延迟)、端到端延迟(跨越完整发布-订阅工作流包括全部密码学操作与网络传播)。比较分析采用开销比率将 ML-DSA 测量归一化相对 ECDSA 基线实现，实现后量子迁移成本量化。对每个指标 M 计算开销比率 R = M{ML-DSA} / M{ECDSA}，超过 1.0 的值表示资源消耗增加。这些比率为向后量子密码学标准转型的资源受限 IoT 环境中的部署规划、容量规模、架构优化提供可操作洞察。

2 实验环境搭建进展

2.1 STM32 开发板硬件平台配置

开发板获取与初步验证 本周启动基于论文第 4 节实验方法论的实际硬件平台搭建工作。已获取 STM32F407VG 开发板，该开发板搭载 ARM Cortex-M4F 微控制器，168MHz 主频配合硬件浮点运算单元，1MB Flash 与 192KB SRAM 配置完全符合实验方法论中定义的资源约束场景。完成开发板硬件连接验证，通过 ST-Link 调试器建立稳定的主机-目标板通信链路，确认 USB 供电与串口通信功能正常运行。硬件平台的顺利获取为后续软件环境部署与基准测试实施奠定物理基础。

调试环境与工具链配置 配置 ARM 开发工具链环境，安装 ARM GCC 编译器 10.3.1 版本及配套 binutils 工具集。建立 OpenOCD 调试服务器连接 ST-Link 调试接口，实现程序烧录、断点调试、寄存器监控等核心开发功能。配置串口终端工具以 115200 波特率建立与目标板的控制台通信，用于测试输出与性能数据采集。工具链环境的完整配置支持从源码编译到目标板部署的完整开发流程，为 pqm4 库移植工作提供必要的基础设施。

2.2 pqm4 密码学库移植工作

源码获取与项目结构分析 从 GitHub 官方仓库获取 pqm4 密码学库源代码，该库专门针对 ARM Cortex-M4 平台提供后量子密码学算法优化实现。完成项目结构初步分析，识别 ML-DSA 算法实现的核心模块：基础多项式运算(poly.c)、NTT 变换(ntt.c)、签名生成验证(sign.c)、随机数生成(randombytes.c)等关键组件。pqm4 库采用模块化设计，支持通过编译标志选择不同参数集(ML-DSA-44/65/87)，为后续基准测试提供灵活配置能力。

编译配置与依赖解决 当前正在进行 pqm4 库到 STM32F407VG 平台的移植适配工作。主要挑战包括将库的通用 ARM Cortex-M4 代码适配到 STM32 具体硬件抽象层，解决随机数生成器接口与 STM32 RNG 外设的对接，配置 DWT 性能计数器以支持实验方法论中定义的周期精度测量。编译系统需要整合 pqm4 提供的 Makefile 与 STM32 HAL 库构建脚本，确保所有依赖组件正确链接。当前已完成基础编译环境配置，正在解决头文件路径与链接器脚本适配问题。

2.3 MQTT 客户端集成准备

Paho MQTT 库评估 开始评估 Eclipse Paho MQTT 嵌入式 C 客户端库在 STM32 平台上的集成可行性。该库提供轻量级 MQTT 3.1.1 协议实现，内存占用与处理开销符合资源受限设备

要求。完成库文档研读，识别关键配置参数：最大消息尺寸(需要容纳 ML-DSA 签名的 2.4-4.6KB 载荷)、网络缓冲区大小、QoS 级别支持等。Paho 库的模块化网络接口设计支持对接 ESP32 WiFi 模块，为 MQTT 通信提供灵活的网络传输层抽象。

ESP32 WiFi 模块对接方案 规划 ESP32-WROOM-32 无线模块与 STM32 主控板的通信方案。ESP32 模块通过 UART 串口与 STM32 建立 AT 指令通信，STM32 发送 AT 指令控制 WiFi 连接、TCP 套接字建立、数据收发等网络操作。该方案将网络协议栈处理负载卸载到 ESP32 模块，STM32 专注于 ML-DSA 密码学运算与 MQTT 应用逻辑，符合实验方法论中定义的性能测量隔离原则。当前正在准备 ESP32 固件烧录与 AT 指令接口验证，计划下周完成 WiFi 模块基础通信功能测试。

2.4 当前进度总结与待解决问题

已完成工作 实验环境搭建工作已完成硬件平台获取与初步验证、开发工具链配置、pqm4 库源码分析等基础性工作。硬件平台的物理连接与调试接口已建立稳定运行状态，为后续软件部署提供可靠基础。编译工具链环境配置完整，支持从源码构建到目标板烧录的完整开发流程。

进行中任务 当前核心任务集中在 pqm4 库的平台移植适配与 MQTT 客户端集成准备。pqm4 库移植工作正在解决 STM32 平台特定的硬件抽象层适配问题，预计需要 1-2 周完成所有编译配置与依赖解决。MQTT 客户端集成处于方案规划阶段，ESP32 WiFi 模块的固件准备与接口验证是下周重点工作。

待解决技术挑战 实验环境搭建过程中识别出若干待解决技术挑战。pqm4 库的随机数生成器需要对接 STM32 硬件 RNG 外设，确保密码学质量的随机性来源。DWT 性能计数器的精确配置与时间戳采集代码需要验证测量准确性，避免引入系统性测量误差。MQTT 载荷嵌入式签名的消息格式设计需要定义签名附加位置、元数据编码方式、密钥标识符格式等协议细节。这些技术问题的解决是下周工作的核心目标，直接影响实验数据采集阶段的顺利开展。