# 【专题2：工业软件的通讯协议】2.Modbus的安全性和可靠性

## 一起来读开源工业软件系列



爱科技勤折腾的苹果二 2023-8-5

# 讨论内容

0 回顾NModbus

1 Modbus的安全性和可靠性

- 1.1 Modbus/TCP Security标准
- 1.2 在Modbus通讯中增强安全性的方法和开源工具
- 1.3 如何Modbus应用中增加安全机制：身份验证、数据加密
- 1.4 保持通信的实时性和稳定性的方法
- 1.5 处理通信中出现的错误和异常的方法

2 查看Modbus安全相关代码

3 参考资料

# 0 回顾上一次的内容

1 前导知识

- 1.1 客户端和服务器
- 1.2 设计模式：Factory、Facade和Adapter
- 1.3 通讯协议：Modbus

2 NModbus

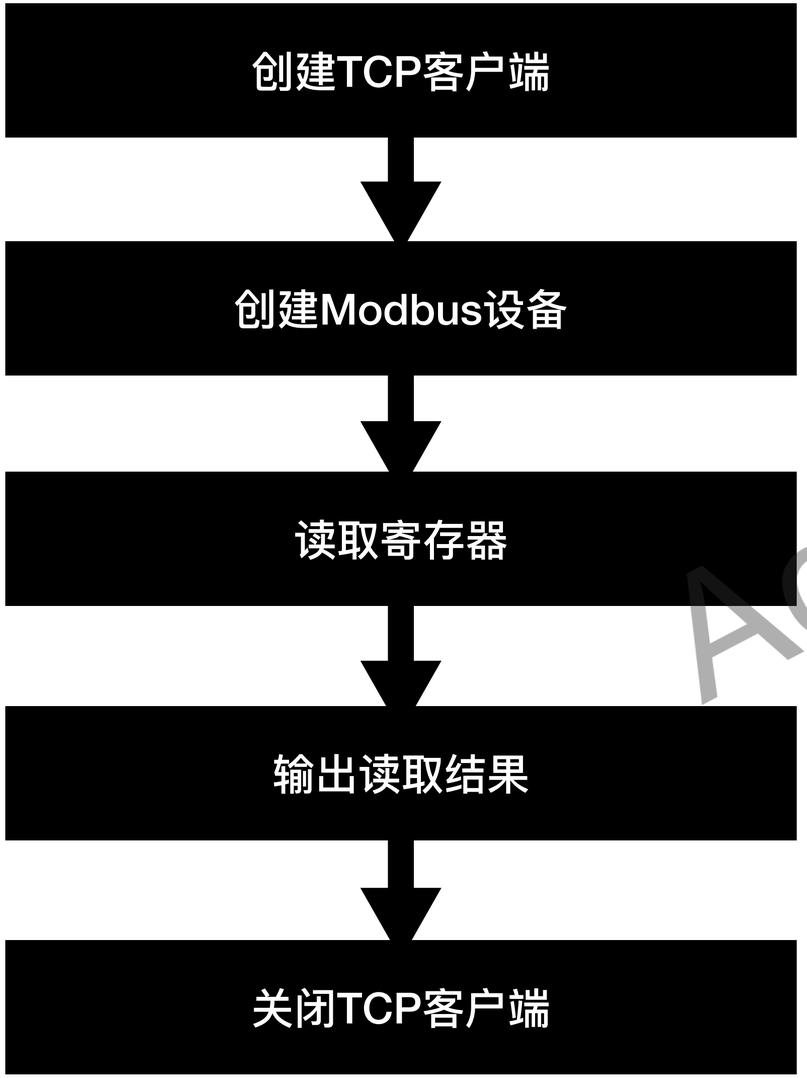
3 阅读NModbus样码

4 参考资料

# 阅读代码注意事项

- 通信方式和通信参数：串口、TCP、以太网和通信速率、超时时间、重试次数
- 数据类型：整型、浮点型、布尔型。
- 读写操作：寄存器地址
- 使用的模块：Data/Device/Extensions/IO/Interfaces/Logging/Message/Utility
- 使用的设计模式：Factory，Facade，Adapter
- 如何处理异常：捕获异常、设置超时、重试机制、日志记录。
- 如何处理错误：地址越界、数据类型不匹配，使用错误码提示用户或操作。

# 使用NModbus库的样码




```
using System;
using System.Net;
using System.Net.Sockets;
using Modbus.Data;
using Modbus.Device;

namespace ModbusExample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 创建TCP客户端
            TcpClient client = new TcpClient("192.168.1.1", 502);

            // 创建Modbus设备
            ModbusIpMaster modbusDevice = ModbusIpMaster.CreateIp(client);

            // 读取寄存器
            ushort startAddress = 0;
            ushort numRegisters = 10;
            ushort[] registers = modbusDevice.ReadHoldingRegisters(1, startAddress,
numRegisters);

            // 输出读取结果
            for (int i = 0; i < numRegisters; i++)
            {
                Console.WriteLine("Register {0}: {1}", startAddress + i, registers[i]);
            }

            // 关闭TCP客户端
            client.Close();
        }
    }
}
```

# 1 Modbus的安全性和可靠性

# Modbus在安全方面存在的问题和解决方案

**问题**

- 无身份验证：Modbus协议本身并不提供任何身份验证机制，这意味着任何可以访问网络的设备都可以发送ModBus命令。
- 无数据加密：Modbus协议并未对数据进行加密，因此，任何人都可以读取或修改传输的数据。
- 无完整性校验：Modbus协议没有提供数据完整性校验机制，因此，数据在传输过程中可能会被篡改。

**解决方案**

- 身份验证：在设备之间建立安全的身份验证机制，只有经过验证的设备才能发送ModBus命令。
- 数据加密：使用安全的加密算法对数据进行加密，保证数据在传输过程中的安全性。
- 完整性校验：通过引入数据完整性校验机制，确保数据在传输过程中的完整性。
- 网络隔离：通过物理或逻辑手段将控制系统网络与其他网络隔离，防止未经授权的访问。
- 定期审计：定期对系统进行审计，检查是否存在安全漏洞，及时进行修复。

# 1.1 Modbus/TCP Security

- 2018 年 8 月，Modbus.org 发布了 Modbus 安全协议https://modbus.org/docs/MB-TCP-Security-v21_2018-07-24.pdf （右图来源） Modbus + TLS + X.509+ACL
- 使用安全协议是确保工业控制系统（ICS）流量安全的基本组成部分。安全协议可以减轻许多常见的网络攻击，包括重放和中间人攻击。
- 通过将传输层安全（**TLS**：众所周知、广为接受的互联网标准）与传统 Modbus 协议相融合来提供强大的保护。TLS 将封装 Modbus 数据包，提供**身份验证和信息完整性保护**。
- **身份验证**：利用 X.509v3 数字证书对服务器和客户端。
- 支持利用 X.509v3 扩展传输基于角色的**访问控制**信息，以授权客户端的请求。
- 将使用一个新端口802，传统的 Modbus 使用端口 502。
- 协议互操作性已于 2017 年 5 月在由四家 Modbus.org 成员公司参加的互操作性活动上进行了演示。利用新协议的产品于2020年上市。
- https://blog.se.com/industry/machine-and-process-management/2018/08/30/modbus-security-new-protocol-to-improve-control-system-security/

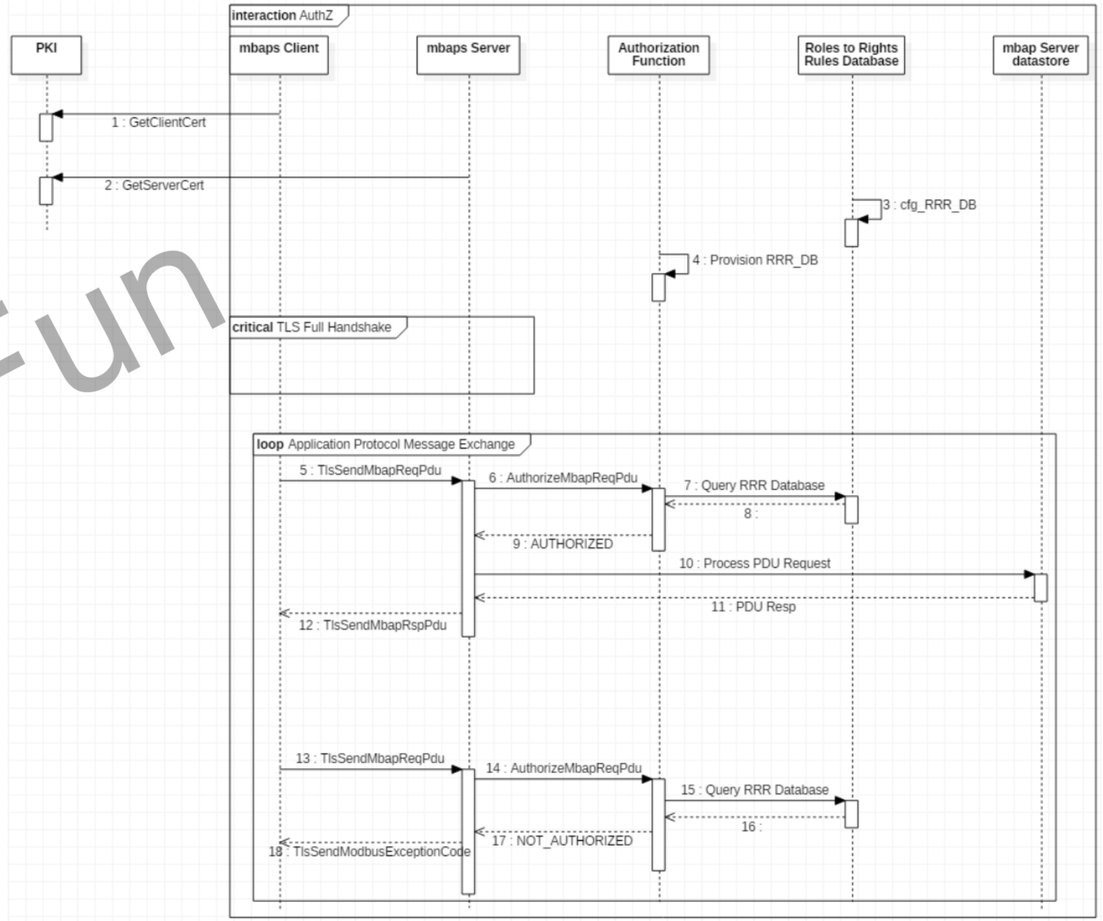
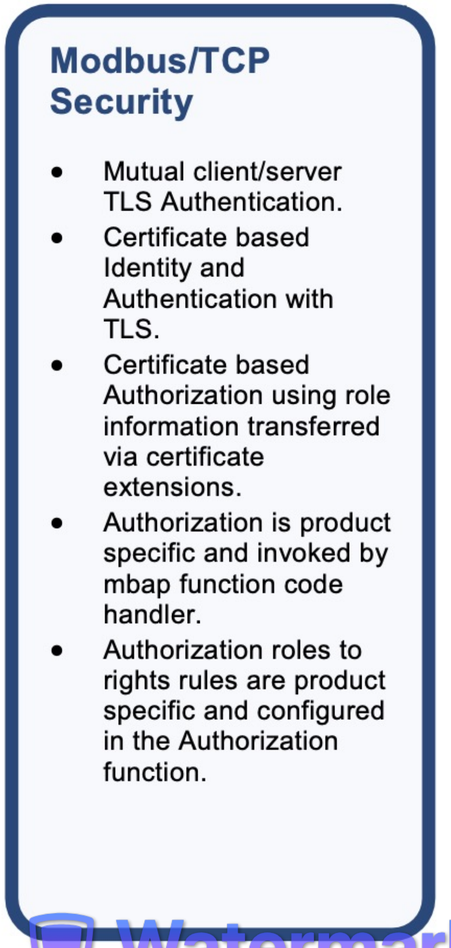



Figure 4 Modbus/TCP Security Concept View

# Modbus/TCP Security（Modbus/TCP 安全性）

- Modbus/TCP 安全性
- 客户端/服务器相互 TLS 验证。
- 基于证书的身份和 TLS 验证。
- 基于证书的授权，使用通过证书扩展传输的角色信息。
- 授权针对特定产品，由 mbap 功能代码处理程序调用。
- 授权角色到权限规则针对具体产品，并在授权功能中配置。








图片来源：https://modbus.org/docs/MB-TCP-Security-v21_2018-07-24.pdf

# 1.2 在Modbus通讯中增强安全性的方法和开源工具

| 名称 | 描述 | 作用 | 代码仓库 |
| --- | --- | --- | --- |
| Wireshark | 网络协议分析器，可以捕获和分析网络上的数据包。 | 可用于检查Modbus通信，看看是否有任何异常的数据包，例如重复的数据包（可能是重放攻击）、大量的错误数据包（可能是拒绝服务攻击）等。https://www.wireshark.org/ | https://github.com/wireshark/wireshark |
| OpenSSL | 强大的安全套接字层密码库，包含了丰富的用于网络安全的功能。 | 可用于创建和管理证书和私钥，用于TLS通信 | https://github.com/openssl/openssl |
| Fail2ban | 防止网络攻击的工具，可以监控系统的日志文件，然后根据日志中的信息，自动屏蔽恶意的IP地址。 | 可防止暴力破解攻击 | https://github.com/fail2ban/fail2ban |
| ModSecurity | 开源的Web应用防火墙，可以防止各种Web攻击。 | 可保护Modbus服务器的Web接口 | https://github.com/SpiderLabs/ModSecurity |
| Snort | 开源的网络入侵检测和防止系统，可以识别和防止各种网络攻击。 | 可监控Modbus通信，防止网络攻击 | https://github.com/snort3/snort3 |

# 1.3 如何Modbus应用中增加安全机制

- 以NModbus为例
  - 可以在发送Modbus命令之前进行**身份验证**，只有验证通过的设备才能发送命令。
  - 可以使用.NET的加密库对**数据**进行**加密**，保证数据在传输过程中的安全性。例如使用SSL/TLS协议。SSL/TLS协议可以提供端到端的数据加密，保证数据在传输过程中的安全。
  - 可以添加**数据完整性校验**，确保数据在传输过程中没有被篡改。

# 1.3.1 实现身份认证的方法：PSK

- 使用预共享密钥（Pre-Shared Key，PSK）。Modbus客户端和服务器都有一个共享的密钥。当客户端发起请求时，它会使用这个密钥对请求进行签名，然后将签名一起发送给服务器。服务器收到请求后，会使用同样的密钥对请求进行验证。如果验证通过，服务器就知道这个请求来自一个合法的客户端。
  - 优点:简单易用，不需要复杂的证书管理。
  - 缺点:
    - 如果密钥被泄露，任何人都可以伪装成合法的客户端。
    - 如果有很多客户端，管理这些密钥可能会很复杂。

# 1.3.2 实现身份认证的方法：数字证书

- Modbus客户端和服务器都有一个数字证书和一个私钥。当客户端发起请求时，它会使用私钥对请求进行签名，然后将签名和证书一起发送给服务器。服务器收到请求后，会使用证书对请求进行验证。如果验证通过，服务器就知道这个请求来自一个合法的客户端。
  - 优点：安全性高，可以防止密钥泄露的问题。
  - 缺点：需要一个可信的证书颁发机构，而且证书的管理可能会比较复杂。

# 1.4 保持通信的实时性和稳定性的方法

- 优化通信参数：可以根据实际情况优化通信参数，如通信速率、超时时间、重试次数等。
- 避免通信冲突：可以避免通信冲突，如避免多个设备同时访问同一寄存器，以防止通信冲突和数据错误。
- 实现数据缓存：可以实现数据缓存，将读取的数据缓存到本地，以减少对设备的访问次数，提高通信的效率和稳定性。
- 在实现数据缓存时，需要注意以下几点，以避免数据的不一致性和错误，确保通信的正确性和稳定性：
  - 同步读写操作：确保读取和写入操作的顺序和时序一致。
  - 锁定缓存区：以防止多个线程同时访问缓存区，导致数据的冲突和错误。
  - 定期刷新缓存：以确保缓存中的数据与设备中的数据一致。
  - 错误处理：如校验和验证等，以确保数据的正确性和可靠性。

# 1.5 处理通信中出现的错误和异常的方法

**日志记录**：使用日志记录工具，记录通信中出现的错误和异常情况，以便进行问题排查和分析。日志记录可以记录通信的时间、数据、错误码等信息，帮助快速定位问题的根源。https://github.com/NModbus/NModbus/tree/develop/NModbus/Logging

**调试工具**：如Modbus调试器等，对通信进行调试和分析。调试工具可以显示通信的数据、错误码等信息，帮助快速定位问题的根源。https://www.modbustools.com/ https://github.com/NModbus/NModbus.DeviceSimulator http://www.plcsimulator.org/

**分析代码**：查找可能导致通信错误和异常的代码段。例如，可能存在的数据类型转换错误、寄存器地址错误等问题。

**设备检查**：检查和测试，以确定设备是否正常工作。例如，可以检查设备的连接状态、电源状态、通信速率等参数。

# 2 查看Modbus安全相关代码

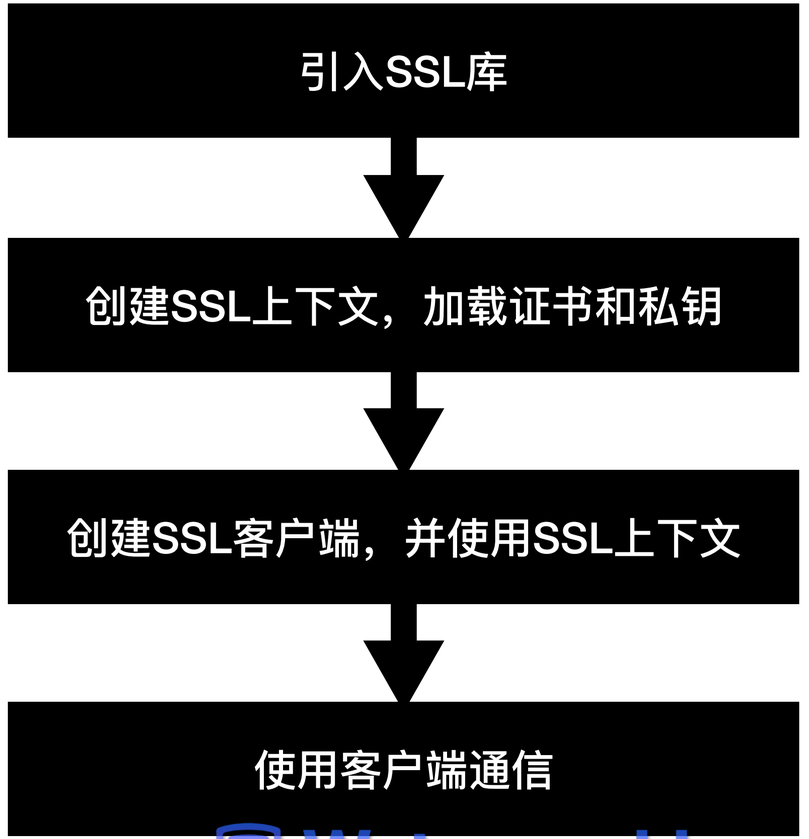
# 使用Pymodbus在应用层实现数据加密

```python
import ssl
from pymodbus.client.sync import ModbusTcpClient

# 创建SSL上下文
context = ssl.create_default_context(ssl.Purpose.CLIENT_AUTH)
context.load_cert_chain(certfile="path_to_certfile", keyfile="path_to_keyfile")

# 创建Modbus客户端
client = ModbusTcpClient('localhost', ssl=context)

# 进行Modbus通信
result = client.read_coils(1,1)
print(result.bits[0])
```




https://pymodbus.readthedocs.io/en/latest/

# 使用Pymodbus实现身份验证：客户端代码

```
import ssl
from pymodbus.client.sync import ModbusTcpClient

ssl_context = ssl.create_default_context(ssl.Purpose.CLIENT_AUTH)
ssl_context.load_cert_chain(certfile='client.crt', keyfile='client.key')
ssl_context.load_verify_locations(cafile='server.crt')

client = ModbusTcpClient('127.0.0.1', ssl_context=ssl_context)  # 连接到服务器
client.write_coil(1, True, unit=1)  # 发送请求，设备地址为1
result = client.read_coils(1, 1, unit=1)  # 读取数据，设备地址为1
client.close()  # 关闭连接
```

# 使用Pymodbus实现身份验证：服务器代码

```py
import ssl
from pymodbus.server.sync import StartTcpServer
from pymodbus.device import ModbusDeviceIdentification
from pymodbus.datastore import ModbusSequentialDataBlock, ModbusSlaveContext, ModbusServerContext

ssl_context = ssl.create_default_context(ssl.Purpose.SERVER_AUTH)
ssl_context.load_cert_chain(certfile='server.crt', keyfile='server.key')
ssl_context.load_verify_locations(cafile='client.crt')

block = ModbusSequentialDataBlock(0, [0]*100)  # 初始化数据块
store = ModbusSlaveContext(di=block, co=block, hr=block, ir=block)  # 初始化数据存储
context = ModbusServerContext(slaves=store, single=True)  # 初始化服务器上下文

identity = ModbusDeviceIdentification()  # 初始化设备身份
identity.VendorName = 'Pymodbus'
identity.ProductCode = 'PM'
identity.VendorUrl = 'http://github.com/riptideio/pymodbus/'
identity.ProductName = 'Pymodbus Server'
identity.ModelName = 'Pymodbus Server'
identity.MajorMinorRevision = '1.0'

StartTcpServer(context, identity=identity, address=('localhost', 5020), ssl_context=ssl_context)  # 启动
```

# TLS

- https://github.com/simonvetter/modbus/blob/master/examples/tls_server.go for TLS and Modbus Security features tls.LoadX509KeyPair
- https://github.com/emelianov/modbus-esp8266/tree/master/examples/TLS Arduino 最完整的 Modbus 库。该库可让 Arduino（阿尔杜伊诺）通过 Modbus 协议进行通信，既可作为主站，也可作为从站或两者兼而有之。支持网络传输（Modbus TCP）和串行线路/RS-485（Modbus RTU）。支持 ESP8266/ESP32 的 Modbus TCP 安全性。
- https://github.com/emelianov/modbus-esp8266/blob/master/examples/TLS/client/client.ino
- https://github.com/emelianov/modbus-esp8266/blob/master/examples/TLS/server/server.ino

# ITI/ICS-Security-Tools

- https://github.com/ITI/ICS-Security-Tools Information Trust Institute The Information Trust Institute (ITI) at the University of Illinois works on research that advances trustworthiness of complex systems.Tools, tips, tricks, and more for exploring ICS Security.

# 3 参考资源

- https://github.com/NModbus/NModbus
- https://github.com/pymodbus-dev/pymodbus
- https://www.simplymodbus.ca/download.htm
- https://www.mesta-automation.com/modbus-with-c-sharp-libraries-examples/
- https://github.com/topics/modbus-tcp-protocol
- https://searchcode.com/codesearch/view/15641076/
- https://asecuritysite.com/subjects/chapter107
- https://www.modbustools.com/
- https://github.com/rossmann-engineering/EasyModbusTCP.NET
- https://www.mdpi.com/1424-8220/22/20/8024 Enhanced Modbus/TCP Security Protocol: Authentication and Authorization Functions Supported

# 下一次，
# C#代码质量

# C#代码质量

- https://www.nuget.org/packages/Microsoft.CodeAnalysis.FxCopAnalyzers
- https://learn.microsoft.com/en-us/visualstudio/code-quality/install-net-analyzers?view=vs-2022
- Code quality rules：https://learn.microsoft.com/en-us/dotnet/fundamentals/code-analysis/quality-rules/