# 通过 WireShark 软件实现 ModbusTcp 报文捕捉与分析 基于 PC 控制 CMMP 伺服



姓名 曹鹏 Festo 技术支持 2020 年 7 月 31 日

### 关键词:

Client, Wireshark, Modbus/TCP, ModbusTcp, 报文捕捉

#### 摘要:

本文介绍如何使用 Wireshark 进行 ModbusTCP 报文捕捉与分析(PC 控制 CMMP 伺服时),文档主要内容包括 ModbusTCP 报文结构介绍,Wireshark 使用介绍,以及报文抓取案例分析等。

#### 目标群体:

本文仅针对有一定自动化设备调试基础的工程师,需要对 ModbusTCP 通讯协议和 Wireshark 软件有一定了解。

#### 声明:

本文档为技术工程师根据官方资料和测试结果编写,旨在指导用户快速上手使用 Festo 产品,如果发现描述与官方正式 出版物冲突,请以正式出版物为准。

我们尽量罗列了实验室测试的软、硬件环境,但现场设备型号可能不同,软件/固件版本可能有差异,请务必在理解文 档内容和确保安全的前提下执行测试。

我们会持续更正和更新文档内容, 恕不另行通知。

1	M	ModbusTCP 介绍	4
1	.1	简介	4
1	. 2	2 ModbusTCP 数据帧	4
	1	1.2.1  报文头 MBAP	4
	1	1.2.2   帧结构 PDU	4
	1	1.2.3 功能码	4
	1	1.2.4 应答格式	4
2	Ν	Wireshark 基本用法	5
2	2.1	1 选择网卡	5
2	2.2	2 窗口介绍及使用技巧	6
	2	2.2.1   过滤器	7
	2.	2.2.2 追踪流	9
	2.	2.2.3 着色规则	10
2	2.3	3 捕获结果	10
2	2.4	4     数据包的大致结构	11
2	2.5	5 三次握手	13
	2.	2.5.1 第一次握手	15
	2.	2.5.2 第二次握手	15
	2.	2.5.3 第三次握手	16
2	2.6	6 四次挥手	16
3	挗	操作过程	17
3	3.1	CMMP 设置	17
3	3.2	2  XF208 设置	18
3	3.3	B Modbus Client 设置	18
3	3.4	进行报文抓取	19
4	丰	其他	20
Z	1.1	FHPP 报文展开	20
Z	1.2	2 内容扩展(西门子 PLC 控制 CMMP)	21
Z	i.3	3 模拟通讯中断	21
5	扑	报文导入导出	21

### 1 ModbusTCP 介绍

### 1.1 简介

Modbus 由 MODICON 公司于 1979 年开发,是一种工业现场总线协议标准。1996 年施耐德公司推出基于以太网 TCP/IP 的 Modbus 协议: ModbusTCP。

Modbus 协议是一项应用层报文传输协议,包括 ASCII、RTU、TCP 三种报文类型。

Modbus 标准协议物理层接口有 RS232、RS422、RS485 和以太网接口,采用 master/slave 方式通信。

IANA (Internet Assigned Numbers Authority, 互联网编号分配管理机构) 给 Modbus 协议赋予 TCP 端口号为 502, 这 是目前在仪表与自动化行业中唯一分配到的端口号。

Modbus 设备可分为主站(Poll)和从站(Slave)。一个网络中主站只有一个,从站可以有多个,主站向各从站发送请求帧,从站给予响应。在使用 TCP 通信时,主站为 Client 端,主动建立连接;从站为 Server 端,等待连接。

#### 1.2 ModbusTCP 数据帧

ModbusTCP 的数据帧可分为两部分: MBAP+PDU。

#### 1.2.1 报文头 MBAP

MBAP 为报文头,长度为7字节,组成如下:

事务处理标识	协议标识	长度	单元标识符
2 字节	2 字节	2 字节	1字节

• 事务处理标识:报文的序列号,一般每次通信之后就要加1以区别不同时间的通信数据报文。

- 协议标识: 00 00 表示 ModbusTCP 协议。
- 长度: 表示接下来的数据长度,单位为字节。
- 单元标识符:可以理解为设备地址。

#### 1.2.2 帧结构 PDU

PDU 由功能码+数据组成。功能码占用1个字节,数据长度不定,由具体功能决定。

#### 1.2.3 功能码

Modbus 的操作对象有四种:线圈、离散输入、输入寄存器、保持寄存器。 线圈:PLC 的输出位,开关量,在 Modbus 中可读可写 离散量:PLC 的输入位,开关量,在 Modbus 中只读 输入寄存器:PLC 中由 I/0 系统提供的 16 位地址寄存器,在 Modbus 中只读 保持寄存器:PLC 中可由应用程序更改的 16 位地址寄存器,在 Modbus 中可读可写

根据控制对象不同, Modbus 的功能码主要有: 0x01: 读线圈 0x05: 写单个线圈 0x0F: 写多个线圈 0x02: 读离散量输入 0x04: 读输入寄存器 0x03: 读保持寄存器 0x06: 写单个保持寄存器 0x10: 写多个保持寄存器 0x17: 读/写多个保持寄存器

#### 1.2.4 应答格式

ModbusTCP 主站发送报文的格式如下: 字节 0: 交易标识符 - 由服务器复制 - 通常为 0 字节 1: 交易标识符 - 由服务器复制 - 通常为 0 字节 2: 协议标识符= 0 字节 3: 协议标识符= 0 字节 4: 长度字段(高字节)= 0(因为所有消息均小于 256) 字节 5: 长度字段(低位字节)=后面的字节数 字节 6: 单元标识符(以前是"从站地址") 字节 7: MODBUS 功能代码 字节8: 根据需要提供数据

ModbusTCP 从站响应报文格式同上,但从字节7开始有如下区别:

从站正常响应:请求功能码+响应数据

从站异常响应:异常功能码+异常码,其中异常功能码即将请求功能码的最高有效位置1,异常码指示差错类型 如下为正常情况下应答示例:

在从站 9	)起始地址 4 处读取 1 个 byte 长度的数据	ĺ
请求	00 00 00 00 00 06 09 03 00 04 00 01	
响应	00 00 00 00 05 09 03 02 00 05	

#### Wireshark 基本用法 2

Wireshark(前称 Ethereal)是一个网络封包分析软件,可以截取各种网络封包,显示网络封包的详细信息; Wireshark 使用 WinPCAP 作为接口,直接与网卡进行数据报文交换; Wireshark 是开源软件,针对不同系统有相应软件版本。

#### 2.1 选择网卡

Wireshark 用于获取机器上的某一块网卡的网络包,当机器上有多块网卡的时候,请首先选择需要监控的那块网卡,如

下图,点击"捕获->选项"出现下面对话框。



弹出下图界面

接口 Microsoft: Local Area Connection* 2 > Microsoft: Wi-Fi > Oracle: VirtualBox Host-Only Network	流里	链路层头 混					
NdisWan Adapter: Local Area Connecti Microsoft: Local Area Connection* 3 NdisWan Adapter: Local Area Connecti NdisWan Adapter: Local Area Connecti Adapter for loopback traffic capture Intel(P) Ethernet Connection 1219. //	on* 11 on* 9 on* 10	Ethernet Ethernet Ethernet Ethernet Ethernet Ethernet BSD loopback Ethernet C	+ 3hapty 默认 默认 默认 默认 默认 默认 默认	缓存 (ME) 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2	监控模 捕获过3 	₽ 160 0 02	
☑ 在所有接口上使用混杂模式 新选择按口的捕获过滤器: ↓ host 192.168.0	83	编写排	該法法				适配器接口管理 管理接口 编译BPFs

点击上图窗口中"管理接口"按钮,在弹出窗口中选择需要显示的网卡,可以减少不必要的选择项。

【管理	金口			$\times$
本地接印	口管道远程	安口		
示 示 りりりりりりりり	易记名称 Local Area Con Wi-Fi VirtualBox Host Local Area Con Local Area Con Local Area Con Local Area Con Adapter for loo Ethernet	接口名称 \Device\NPF_{89078397-1208-49CE-BFBC-F4E9366F1C8D} \Device\NPF_{384D3D9F-2FBA-4404-8B75-62C217A6D20A} \Device\NPF_{AD50B3E3-1711-430B-B866-B9AA5A8B4BAE} \Device\NPF_{AD50B3E3-1711-430B-B866-B9AA5A8B4BAE} \Device\NPF_{8AFC9CC-EBDC-4ED7-9AB4-AC231E6B7E57} \Device\NPF_{BF9AE086-655D-41A4-8139-4358CA40E114} \Device\NPF_{BF9AE086-655D-41A4-8139-4358CA40E114} \Device\NPF_{5A55F555-5C33-4777-B371-2727F896D195} \Device\NPF_{5A55F555-5C33-4777-B371-2727F896D195} \Device\NPF_{091CC1DF-71E8-4175-9644-EFA33EE90705} \Device\NPF_Loopback \Device\NPF_{CBC5B6E4-02C4-432D-BC9D-7985E6A4D4EB}	注释 Microsoft Oracle NdisWan Adapter Microsoft NdisWan Adapter NdisWan Adapter Intel(R) Ethernet Connection 1219-V	
			OK Cancel Help	

### 2.2 窗口介绍及使用技巧

窗口界面简单介绍,如下图,

	▲ 2020.4 开始捕 文件(F)	1.30.pcapng 获编推(E)	束捕	¥9 - 1	<b>捕获进</b> 兆转(G)	<mark>5项设置</mark> 捕获(	<b>-)</b> 4	<del>}析(A)</del>	ŝ	充计(S	) F	电话(Y)	)	无线(W)	工具	Ļ(Т)	帮助(H)				-		×	<
		6 0		X			) 💼	<u></u>	•			Ð												
显示过滤器	ip.src	== 192.168	3.0.1						_													$\times$		+
	No.	Time			Source							Destin	ation				Protocol		Length	Info				^
	6	28 4.217	261		192.1	68.0.1						192.	168	.0.241			ТСР		219	[TCP	Retra	nsmis		
	6	29 4.217	425		192.1	68.0.24	1					192.	168	.0.1			ТСР		54	50773	→ 10	2 [AC		
	6	30 4.217	559		192.1	68.0.24	1					192.	168	.0.1			ТСР		60	[TCP	Dup A	CK 62		
	6	31 4.217	626		192.1	68.0.24	1					192.	168	.0.1			СОТР		61	DT TP	DU (0	) [CO		
封包列表	6	32 4.217	748		192.1	68.0.24	1					192.	168	.0.1			ТСР		61	[TCP	Retra	nsmis		
19 29 9 9 10	6	33 4.229	284		192.1	68.0.1						192.	168	.0.83			Modbus/T	ГСР	66	Qu	ery: ˈ	Trans		
	6	34 4.230	350		192.1	68.0.8	}					192.	168	.0.1			ТСР		60	502 →	4915	2 [AC		
	6	35 4.230	524		192.1	68.0.8	}					192.	168	.0.1			Modbus/T	ГCР	71	Respo	nse: ˈ	Trans		
	6	36 4.234	554		192.1	68.0.24	1					192.	168	.0.1			СОТР		150	DT TP	DU (0	) EOT		
	6	37 4.234	780		192.1	68.0.24	1					192.	168	.0.1			ТСР		150	[TCP	Retra	nsmis		
	6	38 4.235	241		192.1	68.0.24	1					192.	168	.0.1			СОТР		156	DT TP	DU (0	) EOT		
	< 1	20 4 225	200		100 1	<u></u>						100	100	0.1			COTO		450	OT TO	011 (0	507		
	•																					-		
	> Fram	e 1: 60	bytes	on wi	ire (4	80 bit	s), 6	0 by	tes	capt	ured	(48	0 b:	its) or	n inte	rface	\Device	≥\NPF_{CE	3C5B6E4-0	204-43	2D-BC	9D-79	985E6	5A4[
	> Ethe	rnet II,	Src:	PcsCo	ompu_o	d:4c:e	5 (08	:00:	27:с	d:4c	:e5)	, Ds	t:	Siemen	5_9b:c	3:df	(28:63:3	36:9b:c3:	df)					
封包详细信息	> Inte	rnet Pro	tocol	Versi	ion 4,	Src:	192.1	68.0	.241	, Ds	t: 1	92.1	68.0	9.1										
	> Tran	smission	Cont	rol Pr	rotoco	ol, Sro	Port	: 50	773,	Dst	Por	t: 1	02,	Seq: 1	l, Ack	:: 1,	Len: 0							
	<																							>
	0000	28 63	36 9	b c3	df	08 00	27	cd	4c	e5	08	00	45	00										
	0010	00 28	22 0	10 40	00	40 06	95	bd	<b>c</b> Ø	a8	00	f1	c0	a8										
16进制数据	0020	00 01	C6 5	55 00	66	6f b8	b2	25	00	07	2c	5c	50	10										
- ALL PARA	0030	fa f0	1d a	a4 00	00	00 00	00	00	00	00														
地址栏	0 🛛	2020.4.30.pca	ipng														分组	:4801 · 己垦疗	t: 4801 (100.09	b)	i ii	🗶 Def	ault	

Wireshark 窗口界面主要包含如下几个区域:

a. Display Filter(显示过滤器):用于过滤封包列表所需显示的信息

- b. Packet List Pane(封包列表):显示捕获到的封包,有源地址和目标地址,端口号等
- c. Packet Details Pane(封包详细信息): 显示封包中的字段
- d. Dissector Pane(16进制数据)

e. Miscellanous(地址栏,杂项)

#### 2.2.1 过滤器

使用 Wireshark 时,将会得到大量的冗余信息,此时可考虑通过过滤器进行内容筛选。过滤器有两种,其过滤语法并不 完全相同,具体如下。

#### 2.2.1.1 捕获过滤器

捕捉过滤器在抓包前进行设置,决定抓取怎样的数据,以此减少被捕获的数据量;

捕捉过滤器仅支持协议过滤,区别于显示过滤器既支持协议过滤也支持内容过滤。

在"捕获->捕获过滤器"中设置,如下图:

	1 🖉 🛞 📜 🛅	X 🖸 🔍 🖛 i	🔿 😫 斉 👱 📃		3 1		
📕 应用	显示过滤器 <≤tr -/	/>					
🚄 W	/ireshark · 捕获接口	」 捕获选项设	置				×
输	入 輸出 选项						
	流量	链路层头	混杂	Snap长度 (B)	缓存 (MB)	监控模式	捕获过滤器
het		Ethernet		默认	2	·	tcp or s168.0.83
							様式 捕获过滤器 tcp or s168.0.83 管理接口 ▲ Close Help
<							>
$\checkmark$	在所有接口上使用混:	杂模式 	捕获协议	义过滤			管理接口
所注	选择接口的捕获过滤器	: 📘 tcp or src host	192.168.0.83			× •	编译BPFs
						开始 Cl	ose Help

#### 协议过滤语法规则

语法 <b>:</b>	Protocol	Direction	Host(s)	Value	Logical Operations	Other expression
例子 <b>:</b>	tcp	dst	192.168.0.83	502	0r	tcp dst 192.168.0.83

字段详解:

#### Protocol (协议):

可能值: ether, fddi, ip, arp, rarp, decnet, lat, sca, moprc, mopdl, tcp and udp. 如果没指明协议类型,则默认为捕捉所有支持的协议。 注: 在 Wireshark 的 HELP-Manual Pages-Wireshark Filter 中可查询到支持的协议。

#### Direction (方向):

可能值: src, dst, src and dst, src or dst 如果没指明方向,则默认使用 "src or dst" 作为关键字。

#### Host(s):

可能值: net, port, host, portrange. 默认使用"host"关键字。

#### Value:

端口号,可选值。

## Logical Operations (逻辑运算):

可能值: not, and, or.

否("not")具有最高的优先级。或("or")和与("and")具有相同的优先级,运算时从左至右进行。

#### 2.2.1.2 显示过滤器

对捕捉到的数据包依据协议或包的内容进行显示过滤,过滤有两种方式,协议过滤法和内容过滤法。

#### 1). 协议过滤语法

2020.4.30.pcapng

文件 <b>(F)</b>	编辑 <b>(E</b> )	视图(V)	跳转 <mark>(G)</mark>	捕获 <b>(C)</b>	分析 <b>(A)</b>	统计 <b>(S)</b>	电话(Y)	无线 <b>(W)</b>	工具 <b>(T)</b>	帮助(H)
---------------	----------------	-------	---------------------	---------------	---------------	---------------	-------	---------------	---------------	-------

	0	X	्रे   ९ 🦛 🖬	۱	주 🛓 📜									
📕 ip.src ==	■ ip.src == 192.168.0.241 or ip.dst==192.168.0.83 协议过滤法													
No.	Time		Source			Destinatio	on	Protocol						
8	4 0.629343	;	192.168.	0.24	1	192.1	58.0.1	ТСР						
8	5 0.666461		192.168.	0.1		192.10	58.0.241	ТСР						
8	6 0.666462		192.168.	0.1		192.1	58.0.241	ТСР						
Q	7 0 6913/6		102 168	a 2/	1	102 10	58 0 1	COTP						
语法:	Protocol	•	Stringl	•	String2	Comparison operator	Value	Logical Operations	Other expression					
例子:	ip	•	src	•		==	192.168.0.24	or	Ip.dst					

string1 和 string2 是可选的。

依据协议过滤时,可直接通过协议来进行过滤,也能依据协议的属性值进行过滤。

按协议进行过滤: snmp || dns || icmp 显示 SNMP 或 DNS 或 ICMP 封包。

按协议的属性值进行过滤:

ip. addr == 10.1.1.1 显示来源或目的 IP 地址为 10.1.1.1 的封包。

ip. src != 10.1.2.3 or ip. dst != 10.4.5.6 来源不为 10.1.2.3 或者目的不为 10.4.5.6 的封包。

ip. src == 10.230.0.0/16 显示来自 10.230 网段的封包。

tcp.port == 502 显示来源或目的 TCP 端口号为 502 的封包。

tcp. dstport == 502 目的 TCP 端口号为 502 封包。

http.request.method== "POST" 显示 post 请求方式的 http 封包。

http.host == "tracker.lting.com" 显示请求的域名为 tracker.lting.com 的 http 封包。

tcp.flags.syn == 0×02 显示包含 TCP SYN 标志的封包。

#### 2). 内容过滤语法

2020.4.30.pcapng

文件(F) 编辑(E) 视图(V) 跳转(G) 捕获(C) 分析(A) 统计(S) 电话(Y) 无线(W) 工具(T) 帮助(H)

(		🛞 📜 🛅 🗙 🕻	। ९ 🗢 🔿 🖭 🗿 📃 📃 🦉		
	tcp contain	s "http" 内容过滤流	去		
N	о.	Time	Source	Destination	Protocol
	88	0.681434	Siemens_9b:c3:df	Siemens_77:e8:37	PNIO
	89	0.681498	192.168.0.241	192.168.0.1	ТСР
1	90	0.691959	Siemens_77:e8:37	Siemens_9b:c3:df	PNIO
	91	0.707469	192.168.0.1	192.168.0.241	COTP
	92	0.707469	192,168,0,1	192.168.0.241	ТСР

a. 深度字符串匹配

192.168.0.83

contains : Does the protocol, field or slice contain a value 示例:

tcp contains "http" 显示 payload 中包含"http"字符串的 tcp 封包。

#### b. 特定偏移处值的过滤

tcp[20:3] == 47:45:54 /\* 16 进制形式, tcp 头部一般是 20 字节, 这个是对 payload 的前三个字节进行过滤 \*/

http.host[0:4] == "trac"

过滤中函数的使用 (upper、lower)

upper(string-field) - converts a string field to uppercase

lower(string-field) - converts a string field to lowercase

```
示例:
```

upper(http.request.uri) contains "ONLINE"

Wireshark 过滤支持比较运算符、逻辑运算符,内容过滤时还能使用位运算。

如果过滤器的语法是正确的,表达式的背景呈绿色。如果呈红色,说明表达式有误。

### 2.2.2 追踪流

点击每一行时,Wireshark 很智能的在记录前用标线表明了本次会话的记录范围(下图左侧标线范围)。 TCP 请求是"请求->响应"式的,需要查看对应请求的响应时,可以在下图中选取相应行号并点击,由于该数据包是通 过 TCP 协议通讯,只能选择"追踪流(Follow TCP Stream)"。

	10.000000	192.168.0.1	192.168.0.241	TCP	60 102 → 49983 [ACK]	Seq=1 Ack	=1 Win=8192 I	Len=0	
	2 0.000001	192.168.0.1	192.168.0.241	тср	60 [TCP Dup ACK 1#1]	102 → 499	33 [ACK] Seq=	=1 Ack=1 Win=8192	2 Len=0
Γ	3 0.127654	192.168.0.241	192.168.0.83	Modbus/TCP 此处	是基于TCP协议,所以追踪流为	TCP流 CTIPIAM	: 0, Func	: 23: Read Write	e Register
	4 0.128285	192.168.0.83	192.168.0.241	ТСР	Ignore/Unignore Packet(s)	Ctrl+D	26 Win=2990	Len=0	
	5 0.128507	192.168.0.83	192.168.0.241	Modbus/TCP	设罟/取消设罟 时间参考	Ctrl+T	: 0, Func	: 23: Read Write	e Register
	标线 6 0.215351	192.168.0.241	192.168.0.1	COTP	时间平移	Ctrl+Shift+T			
	7 0.215478	192.168.0.241	192.168.0.83	Modbus/TCP	分组注释。	Ctrl+Alt+C	: 0, Func	: 23: Read Write	e Register
	8 0.215556	192.168.0.241	192.168.0.1	ТСР			102 [PSH, /	ACK] Seq=1 Ack=1	Win=63982 Len=102
	90.216257	192.168.0.83	192.168.0.241	TCP	编辑解析的名称		:=51 Win=2990	0 Len=0	
	100.216442	192.168.0.83	192.168.0.241	Modbus/TCP	作为过滤器应用	, ÷	: 0, Func	: 23: Read Write	e Register
	11 0.232890	192.168.0.1	192.168.0.241	COTP	Prepare as Filter	$\rightarrow$ +			
	12 0.232891	192.168.0.1	192.168.0.241	ТСР	对话过滤器	- <b>\</b> +	.9983 [PSH,/	ACK] Seq=1 Ack=10	03 Win=8192 Len=66
	13 0.234215	192.168.0.241	192.168.0.1	ТСР	对话着色		k=67 Win=639	916 Len=0	
<					SCTP	•	· · · · · · · ·		
-	Erame 3: 79 hytes	on wire (632 hits)	79 hytes cantured (632 h	uits) on interfac	追踪流	•	TCP 流	Ctrl+Alt+Shift+T	
	Ethernet II. Src:	: PosCompu od:4c:e5 (0	8:00:27:cd:4c:e5). Dst:	Festo 0c:88:h7 (	复制	+	UDP 流	Ctrl+Alt+Shift+U	
	Internet Protocol	Version 4. Src: 192.	168.0.241. Dst: 192.168.	0.83	11. 13. 244 Jul		TLS 流	Ctrl+Alt+Shift+S	
	Transmission Cont	trol Protocol, Src Por	t: 50249. Dst Port: 502.	Seg: 1. Ack: 1	协议自选项	•	HTTP 流	Ctrl+Alt+Shift+H	
	Modbus/TCP		c. 55245, 550 rorer 562,	, sequity next 1	Decode As		HTTP/2 Strea	m	
,	Modbus				任新窗口显示分组(W)		QUIC Stream		

下图是根据 TCP 流追踪到的报文

_	3 0.127654	192.168.0.241	192.168.0.83	Modbus/TCP	79	Query: Trans:	0; Unit:	0,	Func:	23:	Read W	lrite	Register
	40.128285	192.168.0.83	192.168.0.241	ТСР	60 50	02 → 50249 [ACK]	Seq=1 Ack=2	6 Win	=2990 L	en=0			
	50.128507	192.168.0.83	192.168.0.241	Modbus/TCP	71 Re	esponse: Trans:	0; Unit:	0,	Func:	23:	Read W	Irite	Register
	7 0.215478	192.168.0.241	192.168.0.83	Modbus/TCP	79	Query: Trans:	1; Unit:	0,	Func:	23:	Read W	Irite	Register
	90.216257	192.168.0.83	192.168.0.241	ТСР	60 50	02 → 50249 [ACK]	Seq=18 Ack=	51 Wi	n=2990	Len=@	)		
	100.216442	192.168.0.83	192.168.0.241	Modbus/TCP	71 Re	esponse: Trans:	1; Unit:	0,	Func:	23:	Read W	Irite	Register
	25 0.325564	192.168.0.241	192.168.0.83	Modbus/TCP	79	Query: Trans:	2; Unit:	0,	Func:	23:	Read W	Irite	Register
	26 0.326263	192.168.0.83	192.168.0.241	ТСР	60 50	02 → 50249 [ACK]	Seq=35 Ack=	76 Wi	n=2990	Len=@	3		
	27 0.327451	192.168.0.83	192.168.0.241	Modbus/TCP	71 Re	esponse: Trans:	2; Unit:	0,	Func:	23:	Read W	Irite	Register
	28 0.435402	192.168.0.241	192.168.0.83	Modbus/TCP	79	Query: Trans:	3; Unit:	0,	Func:	23:	Read W	Irite	Register
	29 0.436290	192.168.0.83	192.168.0.241	ТСР	60 50	02 → 50249 [ACK]	Seq=52 Ack=	101 W	in=2990	Len-	=0		
	30 0.437501	192.168.0.83	192.168.0.241	Modbus/TCP	71 Re	esponse: Trans:	3; Unit:	0,	Func:	23:	Read W	Irite	Register
	33 0.544230	192.168.0.241	192.168.0.83	Modbus/TCP	79	Query: Trans:	4; Unit:	0,	Func:	23:	Read W	Irite	Register

#### 2.2.3 着色规则

在菜单"视图-着色规则"下查看,此处用于不同规则报文的显示颜色设置。

名称	过滤器
✓ Bad TCP	tcp.analysis.flags && !tcp.analysis.window_update
✓ HSRP State Change	hsrp.state != 8 && hsrp.state != 16
Spanning Tree Topology Change	stp.type == 0x80
OSPF State Change	ospf.msg != 1
ICMP errors	icmp.type eq 3    icmp.type eq 4    icmp.type eq 5    icmp.type eq 11    icmpv6.type eq 1    icmpv6.type eq 2    icmpv6.type eq 3    icmpv6.type eq 4    icmpv6.type e
ARP ARP	arp
ICMP	icmp    icmpv6
✓ TCP RST	tcp.flags.reset eq 1
SCTP ABORT	sctp.chunk_type eq ABORT
TTL low or unexpected	(! ip.dst == 224.0.0.0/4 && ip.ttl < 5 && !pim && !ospf)    (ip.dst == 224.0.0.0/24 && ip.dst != 224.0.0.251 && ip.ttl != 1 && !(vrrp    carp))
Checksum Errors	eth.fcs.status=="Bad"    ip.checksum.status=="Bad"    tcp.checksum.status=="Bad"    udp.checksum.status=="Bad"    sctp.checksum.status=="Bad"    sctp.checksum.status==
SMB	smb    nbss    nbns    netbios
HTTP HTTP	http    tcp.port == 80    http2
✓ DCERPC	dcerpc
Routing	hsrp    eigrp    ospf    bgp    cdp    vrrp    carp    gvrp    igmp    ismp
TCP SYN/FIN	tcp.flags & 0x02    tcp.flags.fin == 1
✓ TCP	tcp
UDP	udo

#### 2.3 捕获结果

No.	Time	Source	Destination	Protocol	Length I	info	捕捉到的数据,规则不同,颜色不一样				
	2 0.004621	192.168.0.1	192.168.0.241	ТСР	60 1	L02 → 49983 [ACK]	Seq=1 Ack=1 Win=8192 Len=0				
	3 0.004622	192.168.0.1	192.168.0.241	ТСР	60 [	TCP Dup ACK 2#1]	102 → 49983 [ACK] Seq=1 Ack=1 Win=8192 Len=0				
	7 0.238704	192.168.0.1	192.168.0.241	COTP	120 D	OT TPDU (0) EOT					
	8 0.238704	192.168.0.1	192.168.0.241	ТСР	120 [	TCP Retransmissi	on] 102 → 49983 [PSH, ACK] Seq=1 Ack=103 Win=8192 Len=66				
	13 0.259740	192.168.0.1	192.168.0.241	COTP	312 D	OT TPDU (0) EOT					
	14 0.259740	192.168.0.1	192.168.0.241	ТСР	312 [	TCP Retransmissi	on] 102 → 49983 [PSH, ACK] Seq=67 Ack=110 Win=8192 Len=258				
	190.304714	192.168.0.1	192.168.0.241	ТСР	60 1	L02 → 49983 [ACK]	Seq=325 Ack=117 Win=8192 Len=0				
	200.304715	192.168.0.1	192.168.0.241	ТСР	60 [	TCP Dup ACK 19#1	] 102 → 49983 [ACK] Seq=325 Ack=117 Win=8192 Len=0				
> > > >	<pre>&gt; Frame 2: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface \Device\NPF_{CBC5B6E4-02C4-432D-BC9D-7985E6A4D4EB}, id 0 &gt; Ethernet II, Src: Siemens_9b:c3:df (28:63:36:9b:c3:df), Dst: PcsCompu_cd:4c:e5 (08:00:27:cd:4c:e5) &gt; Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.241 &gt; Iransmission Control Protocol, Src Port: 102, Dst Port: 49983, Seg: 1, Ack: 1, Len: 0</pre>										
00	0 08 00 2	7 cd 4c e5 28	3 63 36 9b c3	df 08 00	45 00	·· '·  · (c 6····	- F 不同的协议层数据含义				
00	10 00 28 40	0 1f 00 00 1e	e 06 da 6e c0	a8 00 01	c0 a8	· (@····· ·n··					
00	20 00 f1 00	0 66 c3 3f 00	0 42 66 42 6d	8c de 52	50 10	f.?∙B fBm•	RP 原始数据				
00	30 20 00 9	7 88 00 00 00	00 00 00 00	00							

如果捕获的结果错误包过多, Wireshark 抓包时显示 TCP 校验和错误(Checksum incorrect),但应用层的反应显示报 文的 TCP 校验和没问题,很可能网卡配置->高级->Rx Checksum Offload/Tx Checksum Offload 这两处设置是 Enable, 将其整成 Disable 即可,见下图,

代价是网络性能降低。一般由操作系统的 TCP/IP 协议栈完成 TCP/UDP/IP 校验和的计算工作,这两处设置成 Enable 之后,协议栈不再进行校验和的计算,而是由网卡自己完成。

如果在前述位置没有发现 Rx Checksum Offload/Tx Checksum Offload 项,有两种可能,一种是网卡本身不支持这种功能,另一种是网卡驱动未提供配置项,后一种情形居多。

畵	Device N	lanager					_	$\times$
File	Action	n View Help					_	
<b>(</b>	•	i 🗐 🚺 🖬 🛒	Intel(R) Ethernet C	onnection I219-V F	Properties	×		
	> 🏣 D	isplay adapters	Events	Resources	Power	r Management		^
		rmware uman Interface Douice	General	Advanced	Driver	Details		
		DE ATA/ATAPI controllers naging devices eyboards lemory technology dev lice and other pointing lonitors etwork adapters Cisco AnyConnect Se Intel(R) Dual Band W Intel(R) Ethernet Con VirtualBox Host-Only WAN Miniport (IPV6) WAN Miniport (IPV6) WAN Miniport (IPV6) WAN Miniport (IPV6) WAN Miniport (IPV6) WAN Miniport (IPV6) WAN Miniport (PPPC WAN Miniport (PPPC WAN Miniport (PPPC WAN Miniport (PPPC WAN Miniport (PPPC	The following prop the property you w on the right. Property: Receive Buffers Receive Side So Reduce Speed C Speed & Duplex System Idle Pow TCP Checksum TCP Checksum Transmit Buffers UDP Checksum UDP Checksum Wait for Link Wake on Link So Wake on Pattern	erties are available fr vant to change on the saling Dn Power Down ver Saver Offload (IPv4) Offload (IPv4) Offload (IPv4) Offload (IPv4) offload (IPv4) offload (IPv4) offload (IPv4) offload (IPv6) ettings Packet Match	or this network ada e left, and then sele Value: Disabled	pter. Click ect its value		
	· 🔝 🛛	ther devices			OK	Cancel		
	- 📫 Dr						·	
		rinters						~

### 2.4 数据包的大致结构

此处以主站发送报文为例,

第一行:数据包整体概述

No.	Time	Source	Destination	Protocol	Length Info
	816 4.903118	192.168.0.241	192.168.0.1	COTP	61 DT TPDU (0) [COTP fragment, 0 bytes]
	817 4.903346	192.168.0.241	192.168.0.1	ТСР	61 [TCP Retransmission] 50773 → 102 [PSH, ACK] Seq=5261 Ack=9772 Win=63776 Len=7
	818 4.903524	192.168.0.83	192.168.0.1	TCP	60 502 → 49152 [ACK] Seq=349 Ack=418 Win=2990 Len=0
	819 4.904392	192.168.0.83	192.168.0.1	Modbus/TCP	66 Response: Trans: 25; Unit: 255, Func: 16: Write Multiple Registers
	820 4.905116	Siemens_9b:c3:df	Siemens_77:e8:37	PNIO	60 RTC1, ID:0x8000, Len: 40, Cycle:16384 (Valid,Primary,Ok,Run)
	821 4.921548	Siemens_77:e8:37	Siemens_9b:c3:df	PNIO	60 RTC1, ID:0x8000, Len: 40, Cycle:49152 (Valid,Primary,Ok,Run)
	822 4.923150	192.168.0.1	192.168.0.83	Modbus/TCP	66 Query: Trans: 26; Unit: 255, Func: 3: Read Holding Registers
	823 4.923474	192.168.0.83	192.168.0.1	ТСР	60 502 → 49152 [ACK] Seq=361 Ack=430 Win=2990 Len=0
	824 4.924425	192.168.0.83	192.168.0.1	Modbus/TCP	71 Response: Trans: 26; Unit: 255, Func: 3: Read Holding Registers
	825 4.939329	192.168.0.1	192.168.0.83	Modbus/TCP	75 Query: Trans: 27; Unit: 255, Func: 16: Write Multiple Registers
	826 4.939644	192.168.0.83	192.168.0.1	ТСР	60 502 → 49152 [ACK] Seq=378 Ack=451 Win=2990 Len=0
<					

> Frame 819: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Device\NPF\_{CBC5B6E4-02C4-432D-BC9D-7985E6A4D4EB}, id 0 数据包整体概述
 > Ethernet II, Src: Festo\_ac:88:b7 (00:0e:f0:0c:88:b7), Dst: Siemens\_9b:c3:df (28:63:36:9b:c3:df) 链路层双方的MAC地址

> Internet Protocol Version 4, Src: 192.168.0.83, Dst: 192.168.0.1 网络层, 双方的IP

> Transmission Control Protocol, Src Port: 502, Dst Port: 49152, Seq: 349, Ack: 418, Len: 12 传输层, 双方的端口

> Modbus/TCP ModbusTCP数据帧

> Modbus ModbusTCP帧结构PDU

第二行:链路层详细信息,主要的是双方的 MAC 地址报文头 MBAP

```
> Frame 3: 79 bytes on wire (632 bits), 79 bytes captured (632 bits) on interface \Device\NPF {CBC5B6E4-02C4
Ethernet II, Src: PcsCompu_cd:4c:e5 (08:00:27:cd:4c:e5), Dst: Festo_0c:88:b7 (00:0e:f0:0c:88:b7)

    Destination: Festo 0c:88:b7 (00:0e:f0:0c:88:b7)

       Address: Festo_0c:88:b7 (00:0e:f0:0c:88:b7)
       .... .0. .... .... = LG bit: Globally unique address (factory default)
       .... ...0 .... .... .... = IG bit: Individual address (unicast)
   > Source: PcsCompu_cd:4c:e5 (08:00:27:cd:4c:e5)
    Type: IPv4 (0x0800)
> Internet Protocol Version 4, Src: 192.168.0.241, Dst: 192.168.0.83
> Transmission Control Protocol, Src Port: 50249, Dst Port: 502, Seq: 1, Ack: 1, Len: 25
                                                    •••••E•
     00 0e f0 0c 88 b7 08 00 27 cd 4c e5 08 00 45 00
 0000
                                                    ·A[·@·@· \5····
0010 00 41 5b ed 40 00 40 06 5c 35 c0 a8 00 f1 c0 a8
0020 00 53 c4 49 01 f6 86 18 12 0c 00 00 00 01 50 18
                                                    ·S·I····P·
0030 fe e2 c7 a4 00 00 00 00 00 00 00 13 00 1 00 00
                                                    . . . . . . . .
0040 00 04 00 00 00 04 08 00 00 00 00 00 00 00 00
       指向MAC地址
                         来源MAC地址 表明上层(网络层)用的是IPV4协议
第三行:网络层详细信息,主要的是双方的 IP 地址

    Internet Protocol Version 4, Src: 192.168.0.241, Dst: 192.168.0.83

     0100 .... = Version: 4
     .... 0101 = Header Length: 20 bytes (5)
   > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
     Total Length: 65
     Identification: 0x5bed (23533)
   > Flags: 0x4000, Don't fragment
     Fragment offset: 0
     Time to live: 64
     Protocol: TCP (6) 协议类型
     Header checksum: 0x5c35 [validation disabled]
     [Header checksum status: Unverified]
     Source: 192.168.0.241 来源IP
     Destination: 192.168.0.83 _ 目标IP
      00 0e f0 0c 88 b7 08 00
                                 27 cd 4c e5 08 00 45 00
                                                                       1.1...
0000
       00 41 5b ed 40 00 40 06
                                 5c 35 c0
                                           a8
                                              00
0010
                                                     с0
                                                                 ·@·@·
       00
          53 c4 49 01 f6 86 18
                                                              S-I-
0020
                                 12 0c 00 00 00 01 50 18
                                                                             P
0030 fe e2 c7 a4 00 00 00 00
                                 00 00 00 13 00 17 00 00
0040 00 04 00 00 00 04 08 00
                                 00 00 00 00 00 00 00
```

第四行: 传输层的详细信息,主要的是双方的端口号,下图中 49159 是来源电脑端口号,502 是目标端口号,TCP 会话的每一端都包含一个 32 位(bit)的序列号,当主机开启一个 TCP 会话时,初始序列号是随机的,可能是 0 和 4,294,967,295 之间的任意值,而 Wireshark 默认显示的都是相对序列号/确认号,而不是实际序列号/确认号,相对序 列号/确认号是和 TCP 会话的初始序列号相关联的,因为比起真实序列号/确认号,跟踪更小的相对序列号/确认号会相 对容易一些,下图中 189 就是相对序列号。



ModbusTCP 通讯连接遵从 TCP 连接标准,初始连接时需要通过三次握手建立连接。

一次握手,客户端->服务器:发送标识为 SYN=1、随机产生的客户端序号 seq(发送序号)

二次握手,服务器->客户端:发送标识为 SYN=1、ACK=1、第一步产生的客户端序号 seq+1 (Ack 确认序号)、随机产生 的服务端序号 seq

三次握手,客户端->服务器:发送标识 ACK=1,第二步产生的服务端序号 seq+1 (确认序号),随机产生客户端序列号 seq 图中 ACK 为确认标志位,Ack 为确认序列号

601 4.129249	192.168.0.1	192.168.0.83	TCP	60 49152 → 502 [SYN] Seq=0 Win=8192 Len=0 MSS=1460)一次握手
602 4.131053	192.168.0.83	192.168.0.1	TCP	60 502 → 49152 [SYN, ACK] Seq=0 Ack=1 Win=2924 Len=0 MSS=1450)二次握手
603 4.137203	Siemens_9b:c3:df	Siemens_77:e8:37	PNIO	60 RTC1, ID:0x8000, Len: 40, Cycle:57344 (Valid,Primary,Ok,Run)
604 4.144204	192.168.0.1	192.168.0.83	TCP	60 49152 → 502 [ACK] Seq=1 Ack=1 Win=8192 Len=0)三次握手
605 4.147236	192.168.0.1	192.168.0.241	COTP	130 DT TPDU (0) EOT
606 4.147237	192.168.0.1	192.168.0.241	TCP	130 [TCP Retransmission] 102 → 50773 [PSH, ACK] Seq=7829 Ack=4388 Win=8192 Len=76



标志位对应的功能:

URG: 紧急指针 ( urgent pointer) 有效。

ACK: 确认序号有效。

PSH: 从站应尽快响应主站。

RST: 重建连接。

SYN: 同步序号用来发起一个连接。

FIN: 发端完成发送任务。

窗口大小:用于流量控制。

检验和:检验和覆盖了整个的 TCP 报文段: TCP 首部和 TCP 数据,与 udp 相似需要计算伪首部 TCP 数据包结构及在

Wireshark 中的位置。

报文标志位含义

其中[SYN]意为 SYN 位为1(如果没有,则表示为0)。同理如果[]中有 ACK,表示 ACK 位为1

L	tcp.port==	tcp.port==502											
N	D.	Time	Source	Destination	Protocol	Length Info							
	- 601	4.129249	192.168.0.1	192.168.0.83	TCP	60 49152 → 502 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 SYN=1							
	602	4.131053	192.168.0.83	192.168.0.1	TCP	60 502 → 49152 [SYN, ACK] Seq=0 Ack=1 Win=2924 Len=0 MSS=1450 SYN=1,ACK=1	1						
	604	4.144204	192.168.0.1	192.168.0.83	ТСР	60 49152 → 502 [ACK] Seq=1 Ack=1 Win=8192 Len=0 ACK=1							
	614	4.176195	192.168.0.1	192.168.0.83	Modbus/TCP	75 Query: Trans: 1; Unit: 255, Func: 16: Write Multiple Registers							
	618	4.177678	192.168.0.83	192.168.0.1	ТСР	60 502 → 49152 [ACK] Seq=1 Ack=22 Win=2990 Len=0 ACK=1							
	620	4.177998	192.168.0.83	192.168.0.1	Modbus/TCP	66 Response: Trans: 1; Unit: 255, Func: 16: Write Multiple Registers							
	633	4.229284	192.168.0.1	192.168.0.83	Modbus/TCP	66 Query: Trans: 2; Unit: 255, Func: 3: Read Holding Registers							

#### 2.5.1 第一次握手

客户端→服务器端,客户端发送第一个包,其中 SYN 标志位为 1, ACK=0,发送序列号 seq=0。客户端进入 SYN 发送状态,等待服务器确认。

t	p.port=	=502										
No.		Time	Source	Destination	Protocol	Length	Info					
	601	4.129249	192.168.0.1	192.168.0.83	ТСР	60	49152 -	→ 502	[SYN]	Seq=0 Win=8192	Len=0 MSS=1460	
	602	2 4.131053	192.168.0.83	192.168.0.1	ТСР	60	502 → 4	49152	[SYN,	ACK] Seq=0 Ack=	1 Win=2924 Len=0 MSS=	=1450
	604	4.144204	192.168.0.1	192.168.0.83	ТСР	60	49152 -	→ 502	[ACK]	Seq=1 Ack=1 Win	=8192 Len=0	
<												
>	Frame	601: 60 byte	es on wire (480 bits), 60	oytes captured (480	bits) on inter	face \	Device\	NPF_{	CBC5B6	E4-02C4-432D-BC9	D-7985E6A4D4EB}, id	0
> 1	Ether	net II, Src:	Siemens_9b:c3:df (28:63:3	5:9b:c3:df), Dst: Fe	esto_0c:88:b7 (	(00:0e:	f0:0c:8	38:b7)				
> :	Inter	net Protocol	Version 4, Src: 192.168.0	.1, Dst: 192.168.0.8	83							
~	' Transmission Control Protocol, Src Port: 49152, Dst Port: 502, Seq: 0, Len: 0											
	Sou	urce Port: 49	152									
	Des	stination Por	t: 502									
	[S1	tream index:	1]									
	[тс	CP Segment Le	n: 0]									
	Sec	quence number	: 0 (relative sequence	number) seq=0								
	Sec	quence number	(raw): 238298									
	[Next sequence number: 1 (relative sequence number)]											
	Ac	knowledgment	number: 0									
	Ac	knowledgment	number (raw): 0									
	011	L0 = Hea	der Length: 24 bytes (6)									
	∽ Fla	ags: 0x002 (S	YN)									
		000	<pre> = Reserved: Not set</pre>									
		0	<pre> = Nonce: Not set</pre>									
		0	= Congestion Window Red	uced (CWR): Not set								
		0	= ECN-Echo: Not set									
		0	= Urgent: Not set									
			= Acknowledgment: Not set	et								
	0 = Push: Not set											
	0 = Reset: Not set											
	>	1	1. = Syn: Set SYN=1									
			.0 = Fin: Not set									
		[TCP Flags: ·	·····s·]									
00	00 0	00 0e f0 0c 8	8 b7 28 63 36 9b c3 df 08	00 45 00(	c 6E.							
00	10 0	00 2c 0b 00 00	0 00 1e 06 10 28 c0 a8 00	01 c0 a8 ·,····	• •(•••••							
00	20 0	0 53 c0 00 0	1 f6 00 03 a2 da 00 00 00	00 60 02 · S·····	• •••••••							
00	30 2	0 00 91 ad 00	0 00 02 04  05 b4 00 00		• ••••							

2.5.2 第二次握手

服务器端→客户端,服务器收到这个包后发送第二个包,其中包 SYN、ACK 标志位为 1,发送顺序号 seq=0,接收顺序号 Ack=0+1(此处 0 为一次握手 seq=0),此时服务器进入 SYN 接收状态。 No. Time Source Destination Protocol Length Info

601 4.129249 192.168.0.1 192.168.0.83 TCP 60 49152 → 502 [SYN] Seq=0 Win=8192 Len=0 MSS=1460									
602 4.131053 192.168.0.83 192.168.0.1 TCP 60 502 → 49152 [SYN, ACK] Seq=0 Ack=1 Win=2924 Len=0 MSS=1450									
Transmission Control Protocol, Src Port: 502, Dst Port: 49152, Seq: 0, Ack: 1, Len: 0									
Source Port: 502									
Destination Port: 49152									
[Stream index: 1]									
[TCP Segment Len: 0]									
Sequence number: 0 (relative sequence number)响应seq=0									
Sequence number (raw): 0									
[Next sequence number: 1  (relative sequence number)]确定下一步客户端序列号seq=1									
Acknowledgment number: 1 (relative ack number)									
Acknowledgment number (raw): 238299									
0110 = Header Length: 24 bytes (6)									
<pre> Flags: 0x012 (SYN, ACK) </pre>									
000 = Reserved: Not set									
0 = Nonce: Not set									
0 = Congestion Window Reduced (CWR): Not set									
0 = ECN-Echo: Not set									
ACK=1									
U = PUSN: NOT SET									
>									
[ICF Flags, ·····A··s·]									
Window size value, 2924									
0000 28 63 36 9b c3 df 00 0e f0 0c 88 b7 08 00 45 00 (c6E.									
0010 00 2C 1/ 56 00 00 40 06 e1 d1 C0 a8 00 53 C0 a8 , V $(0$									

### 2.5.3 第三次握手

客户端→服务器,客户端收到服务器传来的包后,向服务器发送第三个包,SYN=0,ACK=1,接收顺序号Ack=0+1(此处0为二次握手 seq=0),发送顺序号 seq=1。此包发送完毕,客户端和服务器进入 ESTABLISHED 建立成功状态,完成三次握手。

1/± ] 。

	tcp.port==502										
N	o. Time	Source	Destination	Protocol	Length	Info					
Γ	601 4.129249	192.168.0.1	192.168.0.83	ТСР	60	49152	→ 502	[SYN]	Seq=0 Win=8192 Le	n=0 MSS=1460	
~	602 4.131053	192.168.0.83	192.168.0.1	ТСР	60	502 →	49152	[SYN,	ACK] Seq=0 Ack=1 N	Win=2924 Len=0 /	MSS=1450
	604 4.144204	192.168.0.1	192.168.0.83	ТСР	60	49152	→ 502	[ACK]	Seq=1 Ack=1 Win=8	192 Len=0	
<											
>	Frame 604: 60 byte	es on wire (480 bits), 60 b	ytes captured (480	bits) on inter	face \	Device	\NPF_{	CBC5B6	E4-02C4-432D-BC9D-	7985E6A4D4EB},	id Ø
>	Ethernet II, Src:	Siemens_9b:c3:df (28:63:36	:9b:c3:df), Dst: Fe	esto_0c:88:b7 (	00:0e:	f0:0c:	88:b7)				
>	Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.83										
~	r Transmission Control Protocol, Src Port: 49152, Dst Port: 502, Seq: 1, Ack: 1, Len: 0										
	Source Port: 49152										
	Destination Port: 502										
	[Stream index:	1]									
	[TCP Segment Le	n: 0]									
	Sequence number	: 1 (relative sequence r	number) <mark>seq=1</mark>								
	Sequence number (raw): 238299										
	[Next sequence	number: 1 (relative sequences of the seq	uence number)]								
	Acknowledgment i	number: 1 (relative ack	number)								
	ACKNOWLEdgment I	number (raw): 1									
	0101 = Head	der Length: 20 Dytes (5)									
	* FIASS: 0X010 (A	- Reconved: Not set									
	000	- Nonco: Not set									
		- Congestion Window Podu	red (CWR): Not cot								
	0	- ECN_Echo: Not set	iceu (CWR): NUC SEL								
	0	= Urgent: Not set									
		= Acknowledgment: Set	ACK-1								
	0	. = Push: Not set	ACK-1								
		<pre>. = Reset: Not set</pre>									
		. = Syn: Not set									
		Ø = Fin: Not set									
	[TCP Flags: •	·····A····]									
6	000 00 0e f0 0c 88	3 b7 28 63 36 9b c3 df 08	00 45 00(	6 · · · · E ·							
6	0010 00 28 0b 01 00	0 00 1e 06 10 2b c0 a8 00	01 c0 a8 ·(·····								
6	020 00 53 c0 00 01	L f6 00 03 a2 db 00 00 00	01 50 10 · S·····	· · · · · · · · P ·							
0	020 20 00 20 50 00	00 00 00 00 00 00 00									

### 2.6 四次挥手

假设 Client 端发起中断连接请求,也就是发送 FIN 报文。Server 端接到 FIN 报文后,意思是说"Client 端没有数据要 发送了",但是如果 Client 端还有数据没有发送完成,则 Server 端不必急着关闭 Socket,可以继续发送数据。所以 Server 端先发送 ACK, "告诉 Client 端,你的请求我收到了,但是我还没准备好,请你继续等我的消息"。这个时候 Client 端就进入 FIN\_WAIT 状态,继续等待 Server 端的 FIN 报文。当 Server 端确定数据已发送完成,则向 Client 端发 送 FIN 报文,"告诉 Client 端,这边数据发完了,准备好关闭连接了"。Client 端收到 FIN 报文后,"就知道可以关闭 连接了,但是他还是不相信网络,怕 Server 端不知道要关闭,所以发送 ACK 后进入 TIME\_WAIT 状态,如果 Server 端没 有收到 ACK, Server 端超时后重传 FIN+ACK 报文段,则 Client 端可以重传,Server 端收到 ACK 后,"就知道可以断开连 接了"。Client 端等待了 2MSL (特定时间单位,即 TIME\_WAIT 时间)后依然没有收到回复,则证明 Server 端已正常关

肉, Clien	t 垢也可以大团连按」,	ICP 连按肌区	件大团亅。	
656 7.764697	192.168.0.83	192.168.0.241	TCP	60 502 → 50249 [ACK] Seq=1140 Ack=1701 Win=2990 Len=0
657 7.764891	192.168.0.83	192.168.0.241	Modbus/TCP	71 Response: Trans: 67; Unit: 0, Func: 23: Read Write Register
680 7.981062	192.168.0.241	192.168.0.83	TCP 四次挥手	54 50249 → 502 [ACK] Seq=1701 Ack=1157 Win=64094 Len=0
742 8.918272	192.168.0.241	192.168.0.83	ТСР	54 50249 → 502 [FIN, ACK] Seq=1701 Ack=1157 Win=64094 Len=0
743 8.919239	192.168.0.83	192.168.0.241	TCP	60 502 → 50249 [ACK] Seq=1157 Ack=1702 Win=2990 Len=0
744 8.919801	192.168.0.83	192.168.0.241	TCP	60 502 → 50249 [FIN, ACK] Seq=1157 Ack=1702 Win=2990 Len=0
745 8.919959	192.168.0.241	192.168.0.83	TCP	54 50249 → 502 [ACK] Seg=1702 Ack=1158 Win=64094 Len=0



四次数据包

第一次挥手: 主动关闭方发送第一个包, 其中 FIN 标志位为 1, 发送顺序号 seq 为 X。

第二次挥手: 被动关闭方收到 FIN 包后发送第二个包, 其中发送顺序号 seq 为 Y, 接收顺序号 Ack 为 X+1。

第三次挥手: 被动关闭方再发送第三个包, 其中 FIN 标志位为 1, 发送顺序号 seq 为 Z, 接收顺序号 Ack 为 Y+1。

第四次挥手: 主动关闭方发送第四个包, 其中发送顺序号为 X, 接收顺序号为 Y。至此, 完成四次挥手。

整个挥手过程发送数据包在一定的时间周期内没有收到相应的 ACK,等待一定的时间,超时之后就认为这个数据包丢 失,就会重新发送。这个等待时间被称为 RTO。

### 3 操作过程

设备软硬件型号及版本如下: 西门子网管型交换机 XF208, 固件版本 V5.2.4 伺服控制器 CMMP-AS-C2-3A-M3, 固件版本 4.0.1501.2.4 CMMP 控制软件: Modbus TCP Client V1.0

本文准备通过 CMMP 控制软件 Modbus TCP Client 走 ModbusTCP 控制 CMMP 伺服控制器,中间通过西门子交换机 XF208 进行端口映射,最终在电脑侧通过 WireShark 实现报文捕捉。

#### 3.1 CMMP 设置

驱动器上 X18 口作为 ModbusTCP 通讯网口, DIN1-DIN8 拨到 OFF; 在应用数据中将控制接口更改为 Modbus/TCP, 在现场 总线中设置 TCP 端口以及超时时间。

白 🎲 元件 🧳	运行模式选择 环境/安装	貞 信息		
🗄 省 CMMP-AS: 1200modbust	控制器	马达	轴	轴传动比 (总):
回 ん 配置	CMMP-AS-C2-3A-M3	EMMS-AS-40-M-TM	用户自定义道装袖	(无限制) 1:1
·····································	控制接口:	Modbus/TCP	-	
一 (三) 马达	使用的工作模式			使用的功能
e 💼 轴			I.	
····· <u>+0</u> 寻零	▶ 定位运行模式			位置表顺序运行
→ 尺寸体系	☑ 寻零模式		1	□ 模拟输入定位
▶ 测量系统	□ 插补完位模式			
白 田 控制器			II'	
[1] 闭环控制	□ 速度模式			□ 飞锯
□	□ 扭矩运行模式		1	🗌 编码器仿真(X11/主轴)
				▽ 飞测

₽	项目 🖉 1200modbustcp			
CMMP-AS: 1200modbust ^	运行参数 系数组 FHPP	+ 编辑器		
	控制器	马达	轴	轴传到
·····································	CMMP-AS-C2-3A-M3	EMMS-AS-40-M-TM	用户自定义道装袖(无限制)	1:1
应用数据		Madhur (TCD		
	拴刺接口:	Modbus/ICP		
		502		
	ici smili.	502		
	超时:	2000 ms	超时:	
回 闭环达制	数据包分类:	□ 激活	数据包分类:	
			IP 地址:	
~~~ 数字输出端			子网撞码:	
→下模拟输入端	网络设置	1	- 兰网关•	
模拟输出端			********	
	一设置 FHPP			
直接运行模式	FHPP SCON 第四位	的意思: 'RDYEN' 取代 'VLC	DAD	
点动模式				

#### 3.2 XF208 设置

交换机需要有端口映射功能,如下在西门子 XF208 上设置监控端口,如下图:

#### SIEMENS

Console Sup	pport <b>=</b> Logout	SIMATIC NET
Power         CPU         Port Status           F         RM         P1         P5           L1         P2         P6           P3         P7         P4		SIMATIC NET Industrial Ethernet Switch SCALANCE XF208 192.168.0.2
XF208	Switch Configuration	
System     System     XF208     Agent     Switch     Ports     Cable Tester	此处勾选, 开启监控端口 Mirroring Enabled Mirrored P 此处是被监控端口 「 Aging Enabled	此处是监控PC端口,即Wireshark所在端口 ort: 3 · Monitor Port: 5 · 」 1,即服务器端 Monitor Barrier Enabled
FDB     ARP Table     ELDP     DCD	Aging Time [st	Enabled Oversize Mode

#### 3.3 Modbus Client 设置

打开 Modbus TCP Client,参数如下图设置, IP 地址使用 CMMP-AS 的 X18 口地址,此处为 192.168.0.83,功能码使用 23,使用 FHPP 标准字节通讯。

Wodbus TCP Client V1.0.0.12
Modbus Server Configuration IP-Adresse: 192.168.0.83 Port: 502 😴 Connect
Modbus Function Code
Read / Write Multiple Registers (Function Code 23 (Ox17)) 🔹
Read / Write Multiple Registers Modbus Function Code: 0x17 Start Address: 0
Transaction Identifier: 🔄 📝 Increment Transaction Identifier with each pack
FHPP (8 Bytes or 4 Modbus Words (Quantity of Registers))
FHPP + FPC (16 Bytes or 8 Modbus Words (Quantity of Registers))
V Poll every 100 🚔 ms Start

### 3.4 进行报文抓取

点击 Connect,此时如果能够建立连接,Connect 图标会变成 Disconnect,如下图,

🚧 Modbus TCP Client V1.0.0.12
Modbus Server Configuration
IP-Adresse: 192. 168. 0. 83 Port: 502 - Disconnect
Modbus Function Code
Read / Write Multiple Registers (Function Code 23 (0x17)) 🔹
Read / Write Multiple Registers Modbus Function Code: 0x17 Start Address: 0
Transaction Identifier: 🚔 📝 Increment Transaction Identifier with each pack
FHPP (8 Bytes or 4 Modbus Words (Quantity of Registers))
FHPP + FPC (16 Bytes or 8 Modbus Words (Quantity of Registers))
V Poll every 100 - ms Start

CMMP 伺服正常供电后,点击 Start 开始进行控制,点击 Enable, Stop, Halt,此时驱动器使能成功,如下图,

Modbus TCP Client V1.0.0.12	
Modbus Server Configuration IP-Adresse: 192.168.0.83 Port: 502 - Disconnect	
Modbus Function Code	
Read / Write Multiple Registers (Function Code 23 (0x17))	
Read / Write Multiple Registers Modbus Function Code: Ox17 Start Address: O Transaction Identifier:	with each pack
FMPP (8 Bytes or 4 Modbus Words (Quantity of Registers))	
FMPP + FPC (16 Bytes or 8 Modbus Words (Quantity of Registers))	
V Poll every 100 - ms Start	
FHPP Out Byte 1: CCON: 0x03 OPM=0: Record Selection B7 B8 B5 B4 B3 B2 B1 B0 OPM2 OPM1 Lock Reset Brake Stop Enable Byte 2: CPOS: 0x01 B7 B8 B5 B4 B3 B2 B1 B0 Clear Teach JogN JogP Hom Start Halt Byte 3: Record set number: 0 0 Byte 4, 5, 6, 7, 8: reserved (0x00)	Modbus Data sent: 00 54 00 00 00 13 00 17 00 00 00 04 00 00 00 04 08 03 01 00 00 00 00 00 Number of packets sen 85 Number of bytes sent: Transaction Identif0x0054 Protocol Identif0x0000 Message Length: 0x0013 Unit Identif10x00 Function Code: 0x17 Start Address Read: 0x0000 Quantity of Registers Read: 0x0004 Start Address Write: 0x0000 Quantity of Registers Write: 0x0004 Byte Count: 0x08 FHPP Out Data: 03 01 00 00 00 00 00 00
FHPP In         Byte 1: SCON: 0x13 OPM=0: Record Selection         B7       B6       B5       B4       B3       B2       B1       B0         OPM2       OPM1       FCT       VLoad       Fault       Warn       Op En       Enabl         Byte 2: SPOS: 0x84       B7       B6       B5       B4       B3       B2       B1       B0         Ref       Still FolEr       Mov       Teach       MC       ACK       Halt         Byte 3: Record set number: 0       B7       B6       B5       B4       B3       B2       B1       B0         Byte 4: RSE: 0x00       E7       B6       B5       B4       B3       B2       B1       B0         Func       XLim       VLim       ECC       RC1         Byte 5, 6, 7, 8: Position in positioning unit: 106186       EC       RC1	Modbus Data received:           00 54 00 00 00 08 00 17 08 13 84 00 00 00 01 9E CA           Number of packets received5           Number of bytes received:           Transaction Identif0x0054           Protocol Identif0x0000           Message Length: 0x000B           Unit Identif10x00           Function Code: 0x17           Byte Count: 0x08           FMPP In Data:           13 84 00 00 00 01 9E CA

使用 Wireshark 抓取报文,如下图,抓取的 8 个控制字(03 01 00 00 00 00 00 00 00 00),将其 16 进制准换成 2 进制,得到第 1 个控制字为 00000011,和图片中 0pEn, Enabl 状态对应,得到第 2 个控制字为 00000001,和上图中 Halt 状态对应,以此类推,可以得到其它报文定义。

状态对应,	以此	类推,可以得到其	它报文定义。								
942 18.5	515272	192.168.0.82	192.168.0.83	Modbus/TC	P 79	Query:	Trans:	84; Unit	: 0, Func:	23: Rea	ad Writ
<											
> Frame 942:	: 79 byt	es on wire (632 bits)	, 79 bytes captured	(632 bits) on i	nterface \D	evice\NPF	-{CBC5B6I	E4-02C4-432	2D-BC9D-7985	E6A4D4EB}	, id 0
> Ethernet 1	[I, Src:	PcsCompu_cd:4c:e5 (@	8:00:27:cd:4c:e5),	Dst: Festo_0c:88	:b7 (00:0e:	f0:0c:88:	b7)				
> Internet F	Protocol	Version 4, Src: 192.	168.0.82, Dst: 192.	168.0.83							
> Transmissi	ion Cont	rol Protocol, Src Por	t: 49341, Dst Port:	502, Seq: 2101,	Ack: 1429,	Len: 25					
<ul> <li>Modbus/TCF</li> </ul>	<b>)</b>										
Transac	tion Ide	ntifier: 84									
Protoco	l Identi	fier: 0									
Length:	19										
Unit Id	entifier	: 0									
✓ Modbus	44 5.00	stion Code, Dead Uni	to Degister (22)								
.001 01	11 = Fur	Numbon: Q	te Register (23)								
Read No	rd Count										
Write R	eference	Number 0									
Write W	ord Cour	it: 4									
Byte Co	unt: 8										
Data: 0	30100000	0000000									
0000 00 00	f0 00 8	8 h7 08 00 27 cd 4c	e5 08 00 15 00 ···	E.							
0010 00 41	39 d5 4	0 00 40 06 7e ec c0	a8 00 52 c0 a8 ·A	9·@·@· ~· ·· · R· ·							
0020 00 53	c0 bd 0	1 f6 83 85 0c b7 00	00 05 95 50 18 ·S	P.							
0030 f9 4e	d2 60 0	<mark>0 00</mark> 00 54  00 00 00	13 00 17 00 00 ·N	••••••							
0040 00 04	00 00 0	0 04 08 <mark>03  01 00 00</mark>	00 00 00 00 <mark>控制字</mark> ・・	•••••							
	- 6	ᇥᇔᆋᇰᇰᆺᆘᆂᆖ	(10.04.00.00	00 00 01 05		++ 10 \H	4.1.12.4.4.4.4.4.4.4.4.4.4.4.4.4.4.4.4.4	ليلا م كل		- A JD-	
问埋,如	下图,	肌取的 8 个状态字	(13 84 00 00)	00 00 01 9E	CA),将:	具 16 迅	制准换	成2进制	,得到第	1 个状态	公子
为 000100	11, 与	图中 VLoad, OpEr	, Enabl 状态对/	应,得到第2	个状态字)	与1000(	)100, <u>+</u>	5图片中]	Ref, MC 쓌	态对应	, 第
5-8 个壮太	(字为)	)0 01 0F CA 收1	主 16 进制准 格 成	10 进生 / 23	前的估为1	06186	与因由	相同		• • • •	
5011八元	3子/30	JO UI 9E CA, 何元	专10 近前推获成	10 近前,行3	り印但ノリエ	.00180,	一」四十	小日円。			
946 18.5	16781	192.168.0.83	192.168.0.82	Modbus/TCP	71 Response	e: Trans:	84; Un:	1 <b>t: 0,</b> Fu	nc: 23: Read	Write Reg	,1ster
<											
> Frame 946:	71 byte	s on wire (568 bits), 7	1 bytes captured (568	3 bits) on interfa	ce \Device\N	PF_{CBC5B	6E4-02C4-4	32D-BC9D-79	85E6A4D4EB},	id Ø	
> Ethernet I	I, Src:	Festo_0c:88:b7 (00:0e:	0:0c:88:b7), Dst: Pcs	Compu_cd:4c:e5 (@	8:00:27:cd:4	c:e5)					

- > Internet Protocol Version 4, Src: 192.168.0.83, Dst: 192.168.0.82
- > Transmission Control Protocol, Src Port: 502, Dst Port: 49341, Seq: 1429, Ack: 2126, Len: 17

✓ Modbus/TCP	
Transaction Identifier: 84	
Protocol Identifier: 0	
Length: 11	
Unit Identifier: 0	
✓ Modbus	
.001 0111 = Function Code: Read Write Register (23)	
[Request Frame: 942]	
[Time from request: 0.001509000 seconds]	
Byte Count: 8	
Data: 1384000000019eca	
0000 08 00 27 cd 4c e5 00 0e f0 0c 88 b7 08 00 45 00	···'·L··· ·····E·
0010 00 39 01 10 00 00 40 06 f7 b9 c0 a8 00 53 c0 a8	·9····@· ·····S··
0020 00 52 01 f6 c0 bd 00 00 05 95 83 85 0c d0 50 10	· R· · · · · · · · · · · · · P·
0030 0b ae 71 59 00 00 00 54 00 00 00 0b 00 17 08 13	• • qY • • T • • • • • •
0040 84 00 00 00 01 9e ca 状态字	

### 4 其他

### 4.1 FHPP 报文展开

文档附件中包含报文分析 EXCEL 表格,必要时可借助其进一步分析伺服 CMMP 控制时的 ModbusTcp 报文。

抓到的控制字报文,截取属于 Modbus TCP 部分,如原始数据 005400000013001700000004000000040803010000000000, 点击查询,便可得到结果如下,

查询原始数据	05400000013003	17000000040	0000004	0803010000	0000000	1.输	入截取至	的Modb	usTCP报	文								
报文拆分	事物处理标识符	协议标识符	长度	单元标识符	功能码	状态字 数量	读状态 字数量 (wor d格 式)	写保持 寄存器 起始地 址	写控制 字数量 (wor d格 式)	控制字 字节数	数据1	数据2	数据3	数据4	数据5	数据6	数据7	数据8
	54	0	13	0	17	0	4	0	4	8	3	1	0	0	0	0	0	0
定义	Modbus	sTCP通讯协议	度验证正	确读/写	多个保持	寄存器												
										1								
							查询			2.点击	查询							

抓到的状态字报文,截取属于 FHPP 报文部分,如原始数据 00540000000b0017081384000000019eca,点击查询,便可得 到结果如下,

响应原始数据	00540	0000000b001	708138	400000001	eca 1.1	入截取到的M	<b>AodbusTCF</b>	报文						
报文拆分	事物处理 标识符	协议标识符	长度	单元标识 符	功能码	状态字 数量	数据1	数据2	数据3	数据4	数据5	数据6	数据7	数据8
	0054	0000	000b	00	17	08	13	84	00	00	00	01	9e	ca
定义		ModbusTCP 通讯协议端 口502	长度 验证 正确		读输入 寄存器									
				响应										
				HPJ <u>/57</u>			2	.点击响应						
			L											

备注:表格只供参考,测试数量有限,不排除有不足。

#### 4.2 内容扩展(西门子 PLC 控制 CMMP)

如下图,当使用西门子 PLC 对 CMMP 进行控制时,抓取报文,报文按照 3 次握手结构建立连接,但挥手结构异于标准结构,西门子直接使用 RST 指令直接结束通讯(RST 指令由西门子 ModbusTCP 标准功能块内部指定使用),此处断开连接 是根据 TCP 提供了异常终止连接的方法,即给对方发送一个复位报文段,一旦发送了复位报文段,发送端所有排队等待 发送的数据都将被丢弃。

No.	Time	Source	Destination	Protocol	Length Info
	3539 14.939349	192.168.0.1	192.168.0.83	Modbus/TCP	66 Query: Trans: 408; Unit: 255, Func: 3: Read Holding Registers
	3542 14.955636	192.168.0.1	192.168.0.83	Modbus/TCP	75 Query: Trans: 409; Unit: 255, Func: 16: Write Multiple Registers
	3547 14.986376	192.168.0.1	192.168.0.83	Modbus/TCP	66 Query: Trans: 410; Unit: 255, Func: 3: Read Holding Registers
	3560 15.007522	192.168.0.1	192.168.0.83	TCP	60 49152 → 502 [ACK] Seq=6766 Ack=5946 Win=8182 Len=0
	3571 15.051660	192.168.0.1	192.168.0.83	Modbus/TCP	75 Query: Trans: 411; Unit: 255, Func: 16: Write Multiple Registers
	3588 15.114381	192.168.0.1	192.168.0.83	ТСР	60 49152 → 502 [ACK] Seq=6787 Ack=5958 Win=8192 Len=0
	3589 15.116363	192.168.0.1	192.168.0.83	Modbus/TCP	66 Query: Trans: 412; Unit: 255, Func: 3: Read Holding Registers
	3608 15.207335	192.168.0.1	192.168.0.83	Modbus/TCP	75 Query: Trans: 413; Unit: 255, Func: 16: Write Multiple Registers
	3641 15.290391	192.168.0.1	192.168.0.83	Modbus/TCP	66 Query: Trans: 414; Unit: 255, Func: 3: Read Holding Registers
	3644 15.311407	192.168.0.1	192.168.0.83	TCP	60 49152 → 502 [ACK] Seq=6832 Ack=6004 Win=8175 Len=0
L	3667 15.355339	192.168.0.1	192.168.0.83	ТСР	60 49152 → 502 [RST, ACK] Seq=6832 Ack=6004 Win=8192 Len=0

Transmission Control Protocol, Src Port: 49152, Dst Port: 502, Seq: 6832, Ack: 6004, Len: 0
Source Port: 49152
Destination Port: 502
[Stream index: 1]
[TCP Segment Len: 0]
Sequence number: 6832 (relative sequence number)
Sequence number (raw): 245130
[Next sequence number: 6832 (relative sequence number)]
Acknowledgment number: 6004 (relative ack number)
Acknowledgment number (raw): 6004
<u>0101 = Header Lengt</u> h: 20 bytes (5)
> Flags: 0x014 (RST, ACK)
Window size value: 8192
[Calculated window size: 8192]
[Window size scaling factor: -2 (no window scaling used)]
Checksum: 0x7733 [unverified]
[Checksum Status: Unverified]
Urgent pointer: 0
> [Timestamps]
0030 20 00 77 33 00 00 00 00 00 00 00 00 00 00 ········

#### 4.3 模拟通讯中断

模拟客户端通讯突然中断,此处通过拔取网线,实现物理中断,使用 Wireshark 抓取报文,可以发现服务器会重复发送报文 PSH 指令。服务器把 PSH 置 1,并立即创建一个报文段发往客户端,客户端收到 PSH=1 的报文段,会尽快响应服务器(不再等到整个接收缓存都填满),下图中连续发送两次 PSH 指令而得不到响应,基本可以判断通讯发生过中断。

1000 26.734157	192.168.0.254	192.168.0.83	ТСР	129 [TCP Retransmission]	49906 → 502	[PSH, ACK]	Seq=4451 Ack=3027	7 Win=64417 Len=75
999 26.733976	192.168.0.254	192.168.0.83	ТСР	129 [TCP Retransmission]	49906 → 502	[PSH, ACK]	Seq=4451 Ack=302	/ Win=64417 Len=75

### 5 报文导入导出

报文抓取结束后,可以通过"另存为"保存当前抓取的报文,备份或供其它用途;当需要重新分析报文时,直接双击该导出的文档便可重新打开。如下图,保存格式为.pcapng,如果需要导入报文,直接双击文件打开。

🚄 2020.5.7.pcapng								-		
文件(F) 编辑(E)	视图(V) 跳转((	5) 捕获(C)	分析(A) 统计(	(S) 电话(Y)	无线 <mark>(W)</mark>	工具 <b>(T)</b>	帮助(H)			
Open Ctrl+O Open Recent •		0	🖭 🗿 👤 📃 🔍 Q. Q. 🎹							
		•							010101	-
合并(M)				Desti	nation		Protocol		011010	
从 Hex 转储导/	<b>∖(I)</b>		1	192	.168.0.8	3	ТСР		011100	
Close	Ctrl+	W	1	192	.168.0.2	41	ТСР		011100	1
保存(S)	Ctrl+	S S	1	192	.168.0.2	41	ТСР			
另存为(A)	Ctrl+	Shift+S	1	192	.168.0.8	3	Modbus/T		2020.5.7.pca	png