

实 验 报 告

**实验名称： SNMP协议工作原理验证与分析**

**姓 名： 王俊**

**学 号： 202108060918**

**实验日期： 2024.4.28**

**指导老师： 包博文**

一、实验目的

1、学习捕获SNMP报文。

2、通过分析SNMP报文理解SNMP协议的工作过程、SNMP的报文结构、BER编码及其ASN.1描述。

二、实验内容

1、分析并验证SNMP协议的工作过程；

2、分析并验证SNMP协议数据单元的格式；

3、分析理解协议数据单元及其ASN.1描述。

三、实验工具及环境

已配置了SNMP服务的代理Agent、运行Windows系统的我PC，数据包捕获软件wireshark和snmputil。

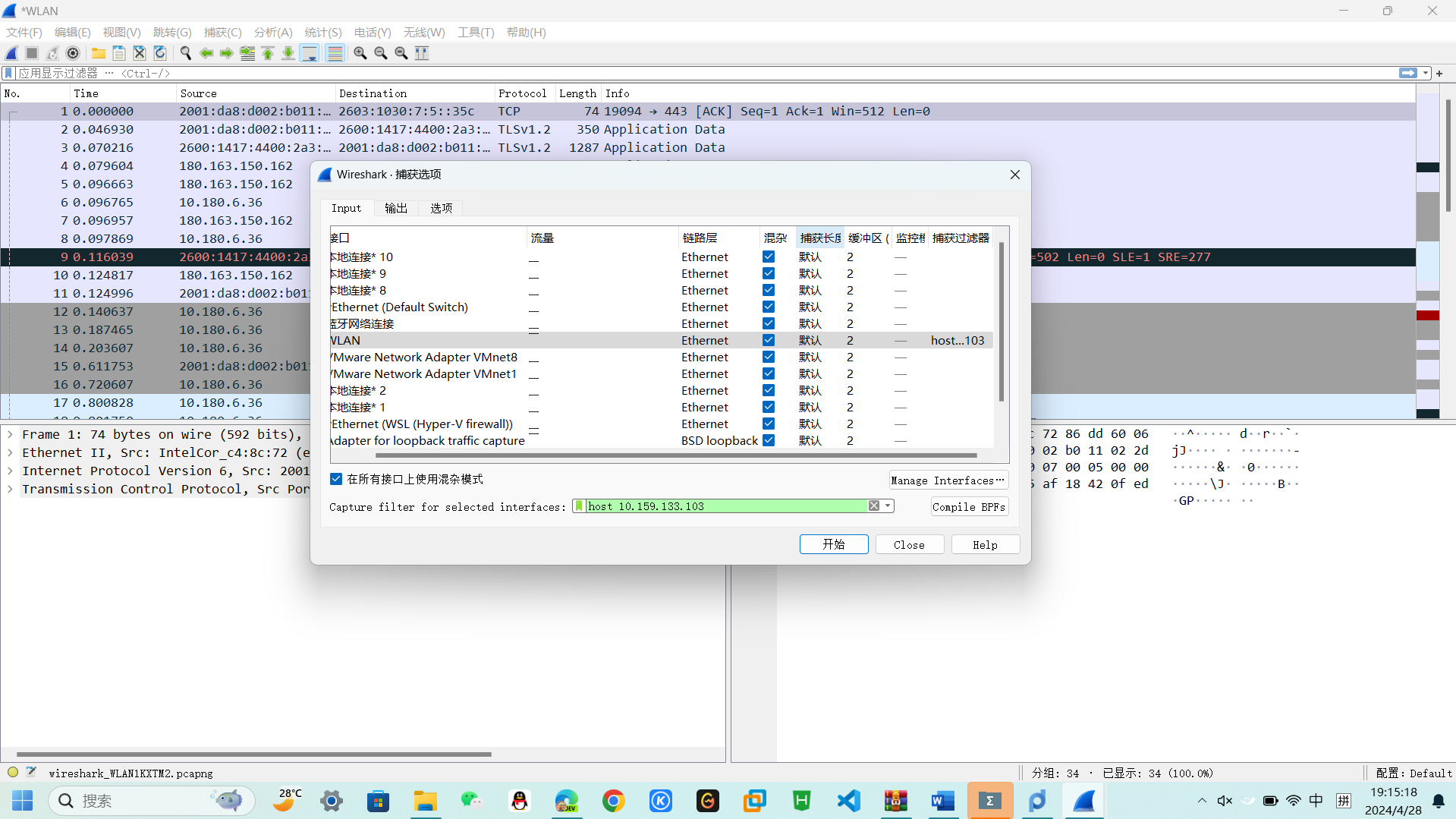
PC1:10.80.6.36

PC2:10.159.133.103

四、实验步骤（截图需配上图标题）

1、分别打开软件wireshark；

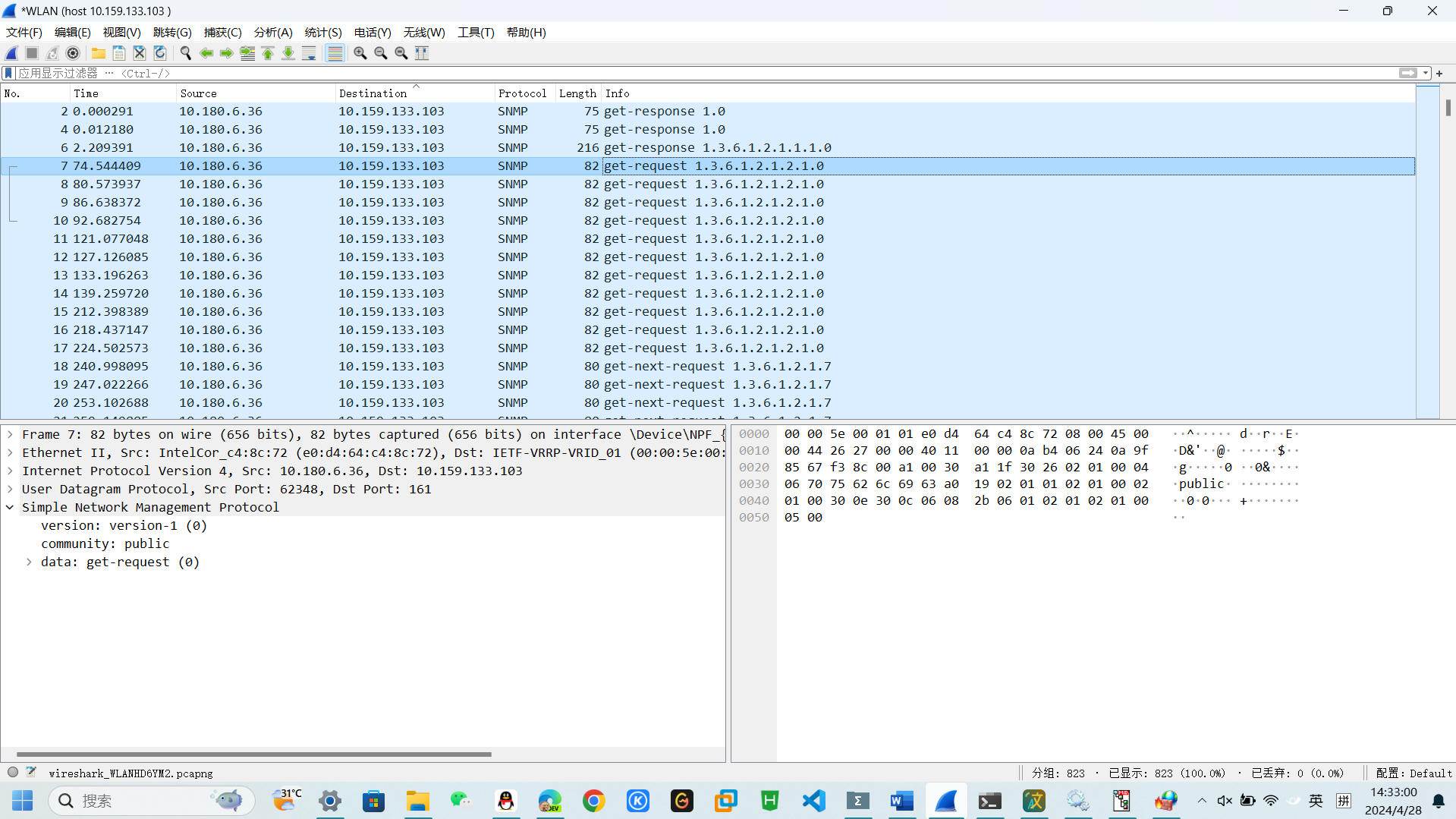
2、首先运行wireshark，设置捕获报文的过滤条件，将其设置为只捕获管理站和代理之间的SNMP报文。



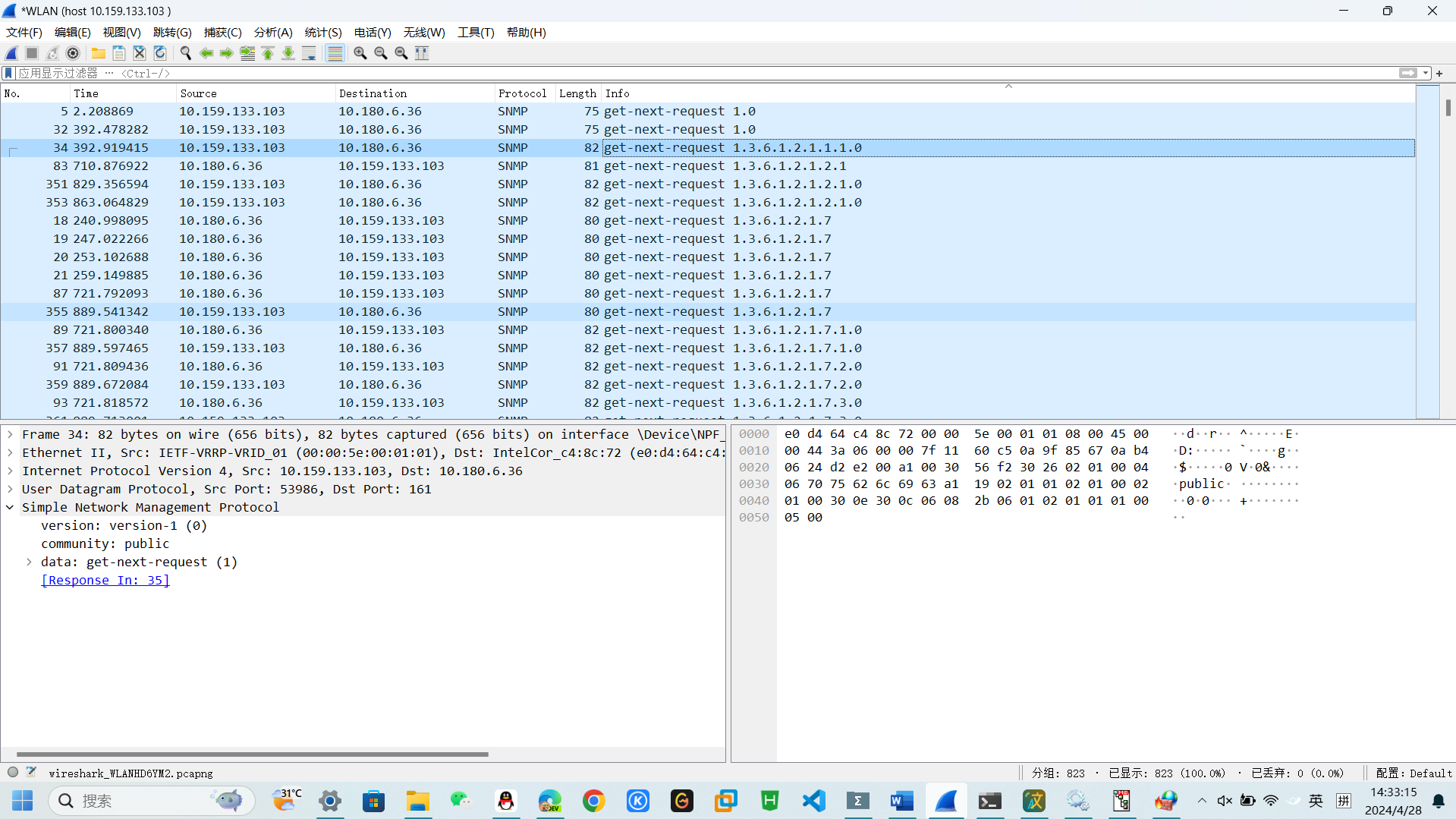
3、点击wireshark中工具栏的start capture，开始捕获SNMP报文；

4、用snmputil访问MIB管理对象实例，分别执行snmputil get、snmputil getNext和snmputil walk，然后观察wireshark内容面板中显示的信息。单击任一信息，右下侧将显示详细的报文数据；

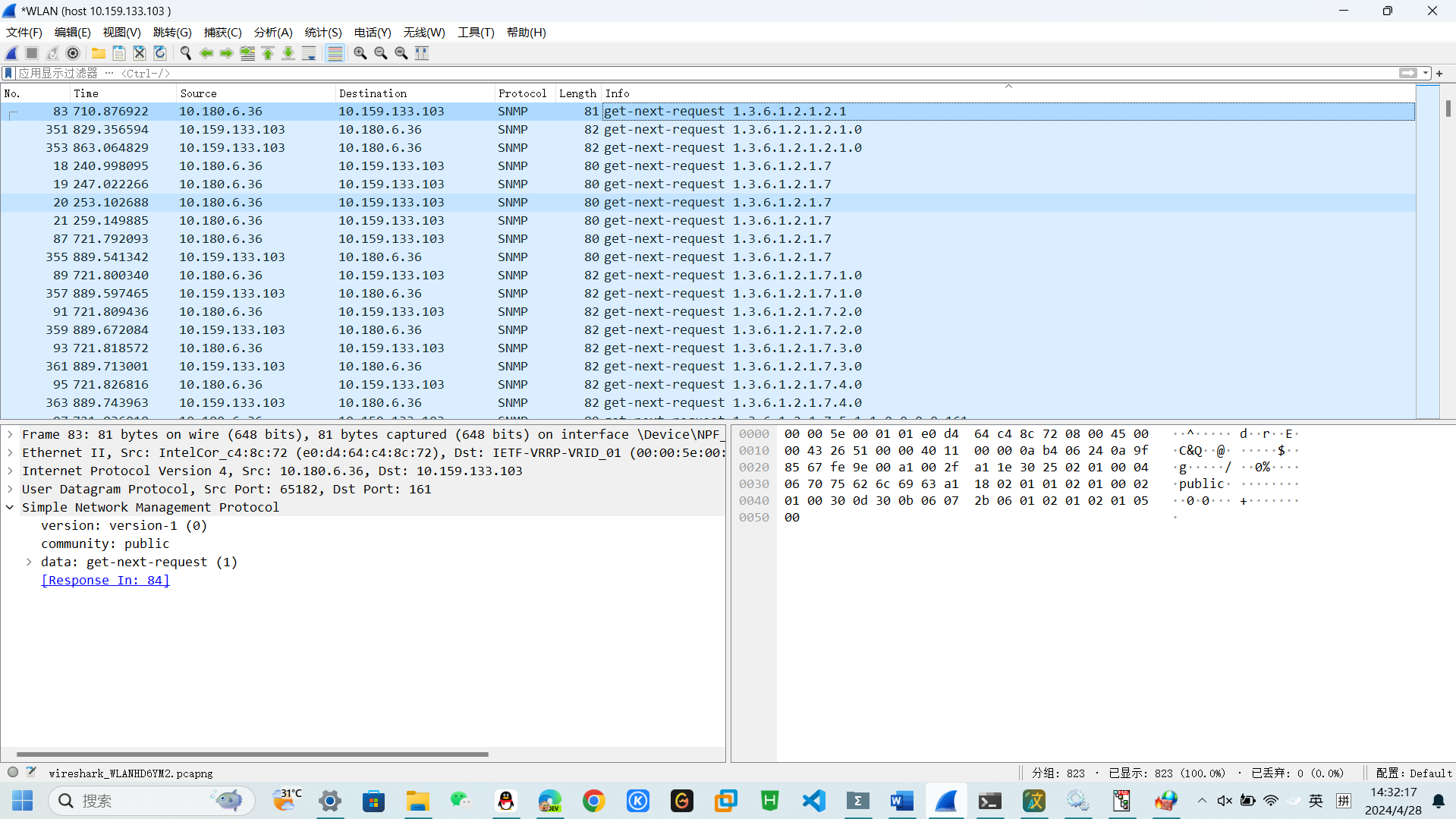
Snmputil Get:



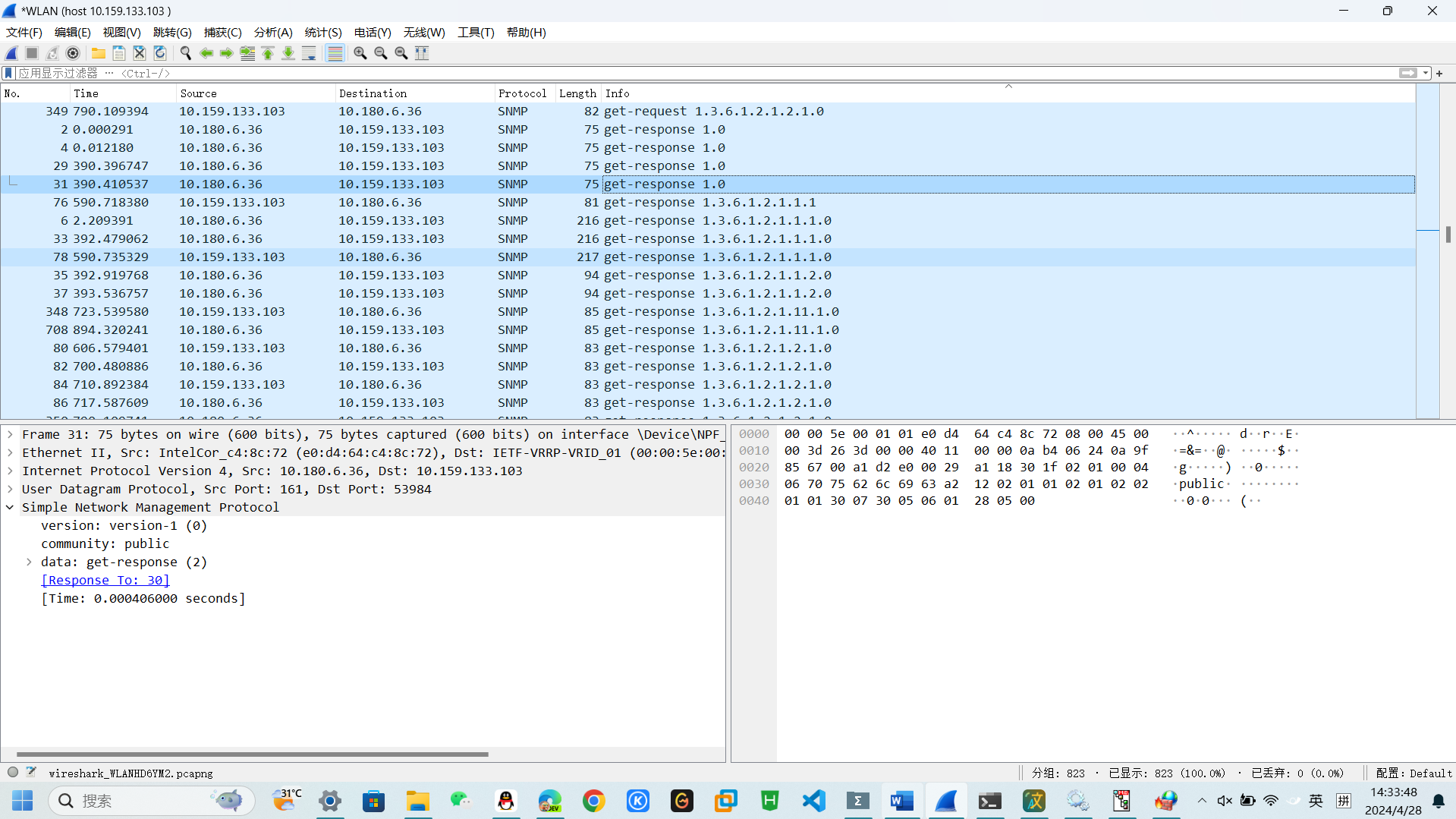
Snmputil GetNext:



Snmputil walk:

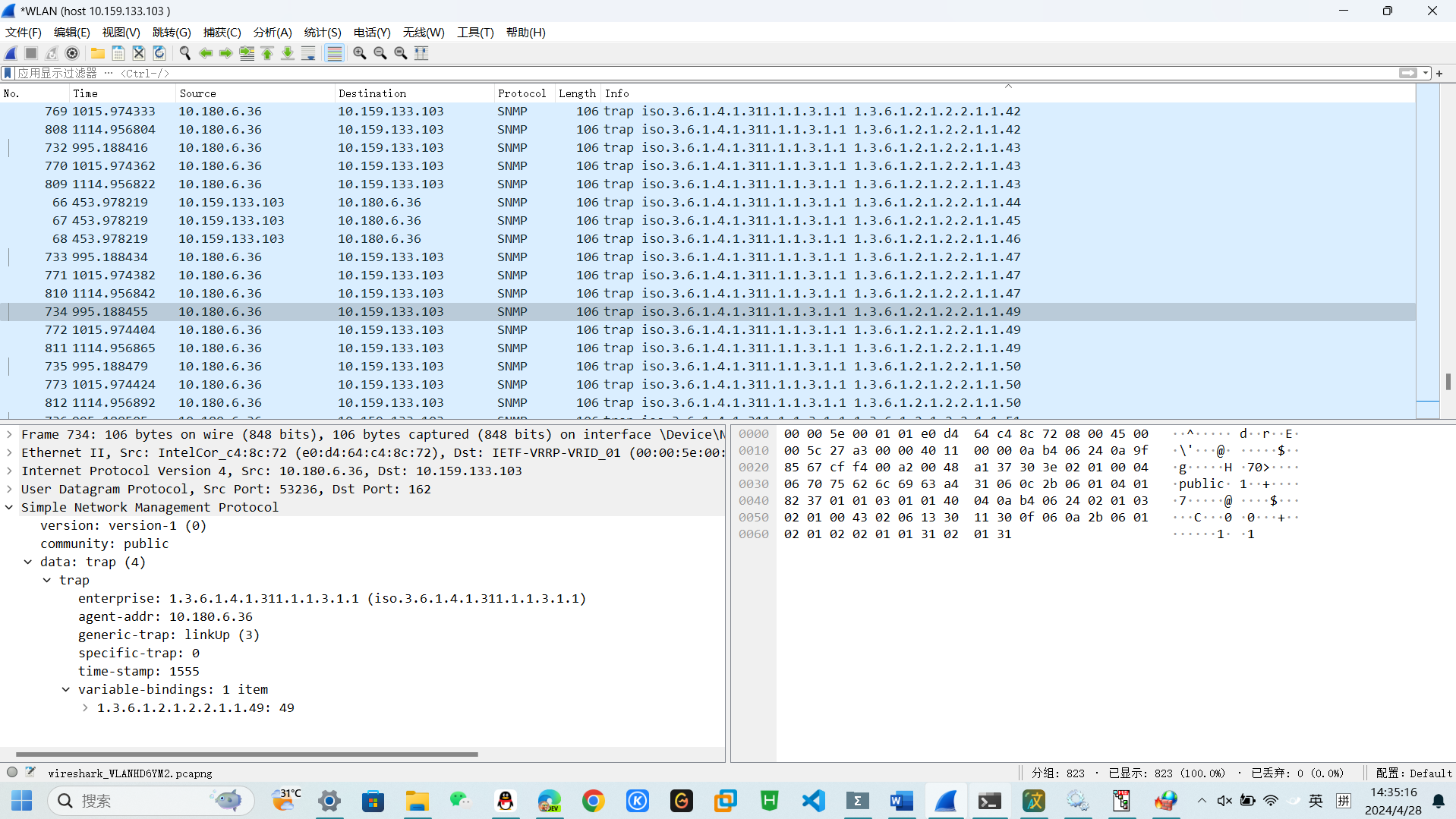


Get Response:



5、使用一错误的团体名（将产生一陷入报文），然后观察wireshark所截获的陷入报文。

Snmputil trap:



五、实验报告

1、分析捕获SNMP报文

* 写出管理站和代理之间的SNMP通信情况，验证SNMP协议的工作过程。

**一、SNMP通信情况概述**

SNMP使用Client/Server（代理/管理）站模型对网络进行管理与维护。每个SNMP代理负责回答SNMP管理工作站（主代理）关于SNMP MIB（管理信息库）定义信息的各种查询。SNMP代理和管理站通过SNMP协议中的标准消息进行通信，每个消息都是一个单独的数据报。在这个过程中，SNMP使用UDP（用户数据报协议）作为第四层协议（传输协议）进行无连接操作。

**二、SNMP协议工作过程**

SNMP（简单网络管理协议）协议的工作过程主要基于客户端-服务器模型，其中SNMP代理充当服务器角色，管理站则充当客户端角色。以下是SNMP协议的基本工作过程：

1.建立连接：

网络设备作为SNMP代理，通过SNMP协议与管理站建立连接。

2.发送请求：

管理站发送SNMP请求消息给SNMP代理，请求获取或修改设备的管理信息。这些请求可能包括设备的状态、配置、性能等。

3.代理处理与响应：

SNMP代理接收请求消息后，根据请求类型执行相应的操作。这可能包括读取MIB（管理信息库）中的信息、执行特定的设备操作等。

执行完请求的操作后，SNMP代理形成响应报文，编码后发送回管理站。

4.接收并处理响应：

管理站接收到来自SNMP代理的响应消息后，进行解码并解析其中的信息。

管理站根据解析得到的信息进行相应的处理，如更新设备状态、展示给管理员等。

5.周期性监视：

如果需要周期性地监视设备状态或配置，管理站可以定期发送SNMP请求消息给SNMP代理，以获取设备的实时状态。

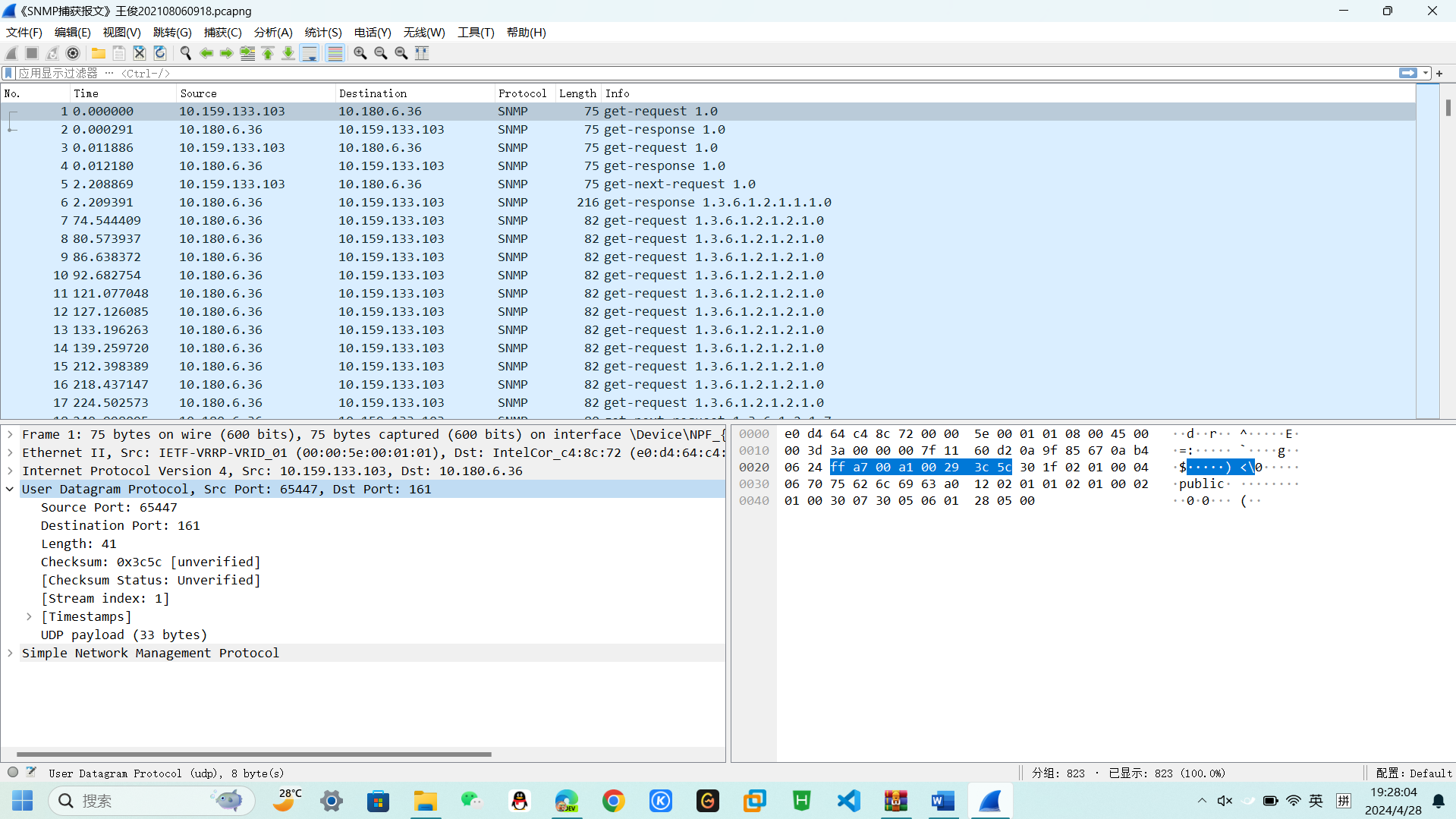
在整个工作过程中，SNMP协议使用UDP（用户数据报协议）进行通信，这是因为UDP是无连接的，适用于网络管理这种可能涉及大量设备的场景。此外，SNMP协议定义了标准的管理信息库（MIB），用于描述和组织设备的管理信息，这使得不同厂商的设备可以通过SNMP进行统一的管理。

**三、验证SNMP协议工作过程**

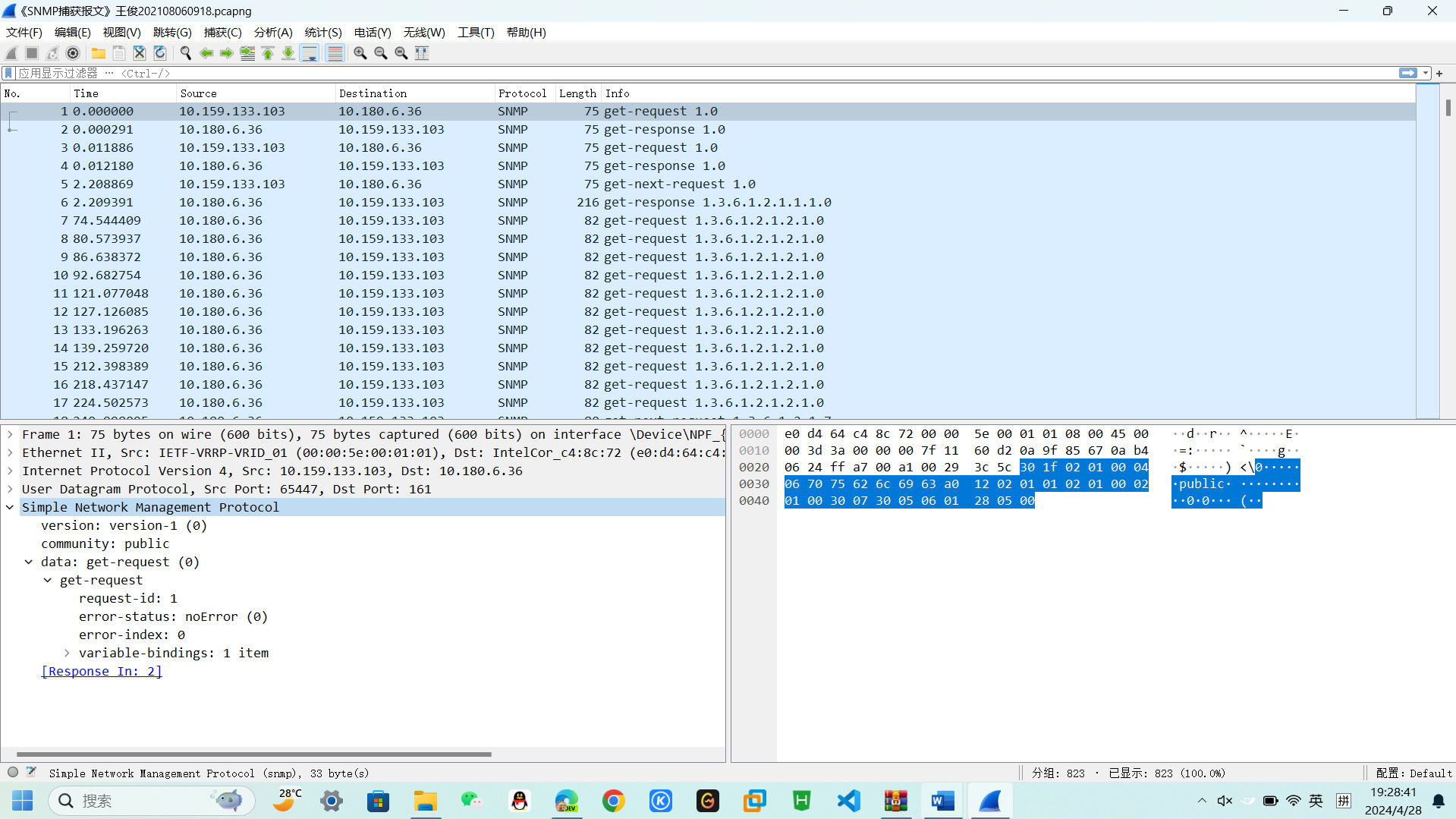
1. **准备环境**：
   * 确保管理站和SNMP代理都已正确配置并运行。管理站通常运行SNMP管理软件，而SNMP代理则部署在网络设备上。
   * 确定管理站和SNMP代理之间的网络连接正常，并且它们之间的通信端口（通常是UDP 161端口）没有被防火墙等安全设备阻塞。
2. **使用SNMP管理工具**：
   * 使用SNMP管理工具（如snmpget、snmpwalk等）来发送SNMP请求到SNMP代理。这些工具允许用户指定要查询的OID（对象标识符），以获取特定的管理信息。
   * 根据SNMP协议的规定，发送请求消息到SNMP代理的UDP 161端口。
3. **观察SNMP通信**：
   * 使用网络抓包工具（如Wireshark）来捕获管理站和SNMP代理之间的通信数据包。这有助于观察SNMP协议的实际工作过程和消息格式。
   * 分析捕获的数据包，确认NMS发送的请求消息和SNMP代理发送的响应消息的内容是否正确，以及它们之间的交互是否符合SNMP协议的规定。
4. **验证响应内容**：
   * 检查SNMP代理返回的响应消息是否包含了请求的管理信息，并且这些信息是否准确反映了网络设备的当前状态或配置。
   * 对比SNMP管理工具的输出结果和网络设备的实际状态，验证SNMP协议工作过程的正确性和可靠性。
5. **分析错误和异常**：
   * 如果在验证过程中遇到错误或异常，仔细分析原因，并检查管理站、SNMP代理的配置以及网络环境是否存在问题。
   * 根据分析结果调整配置或解决网络问题，然后重新进行验证。

* 比较捕获的陷入报文、请求报文和应答报文。分析他们的UDP头部、SNMP报文头部的区别

请求报文：UDP:

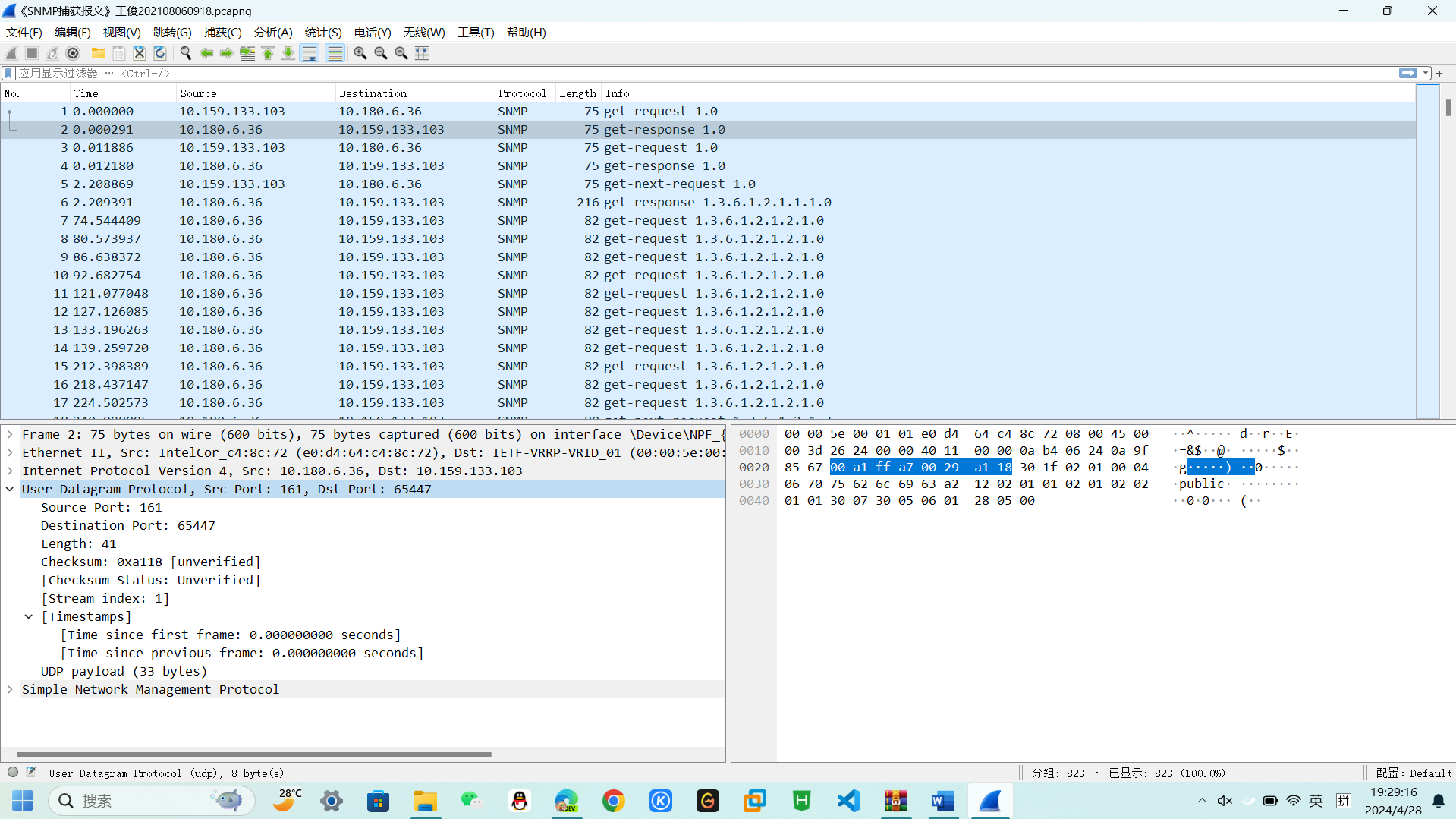


SNMP:

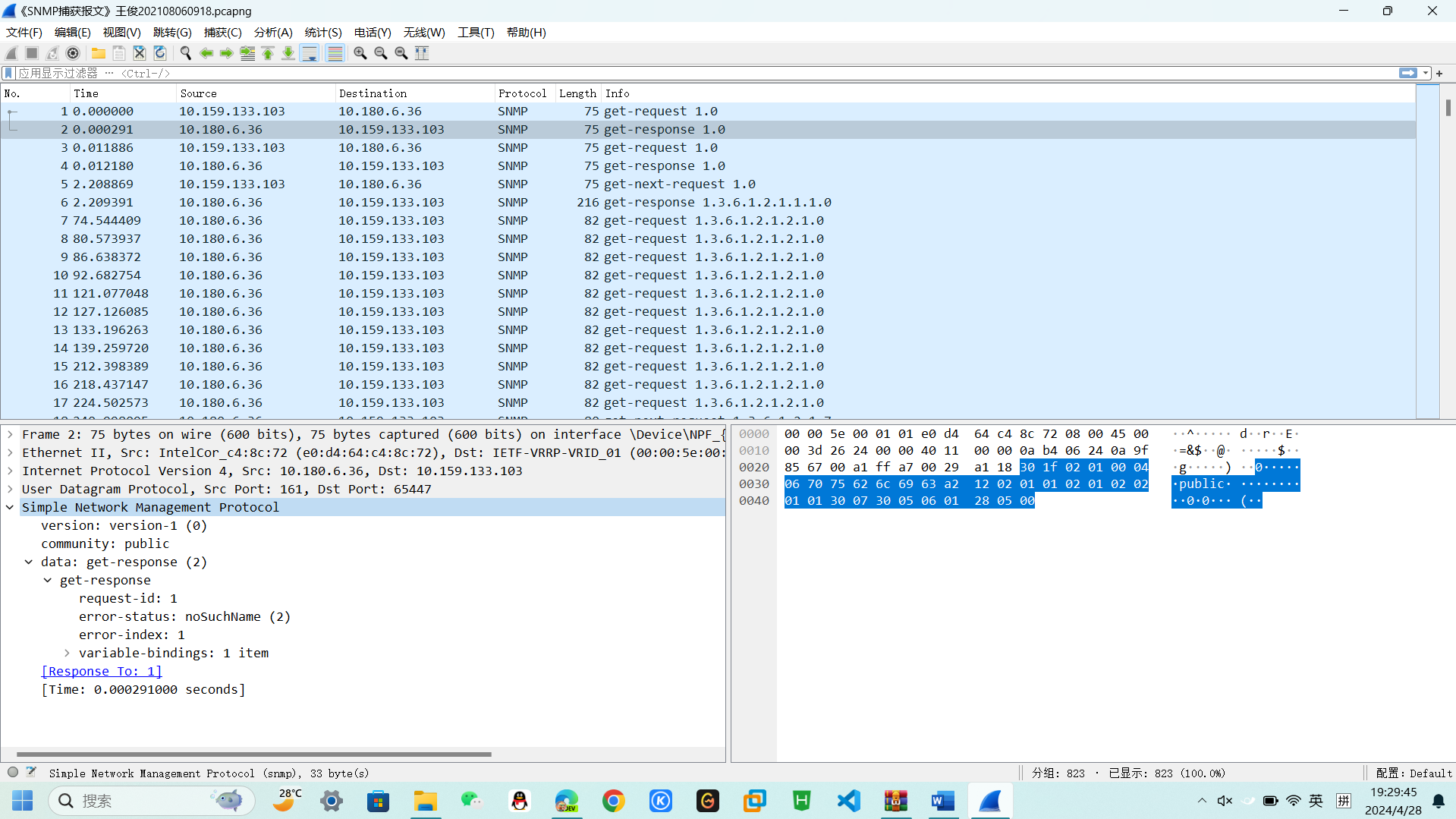


请求报文：

UDP:

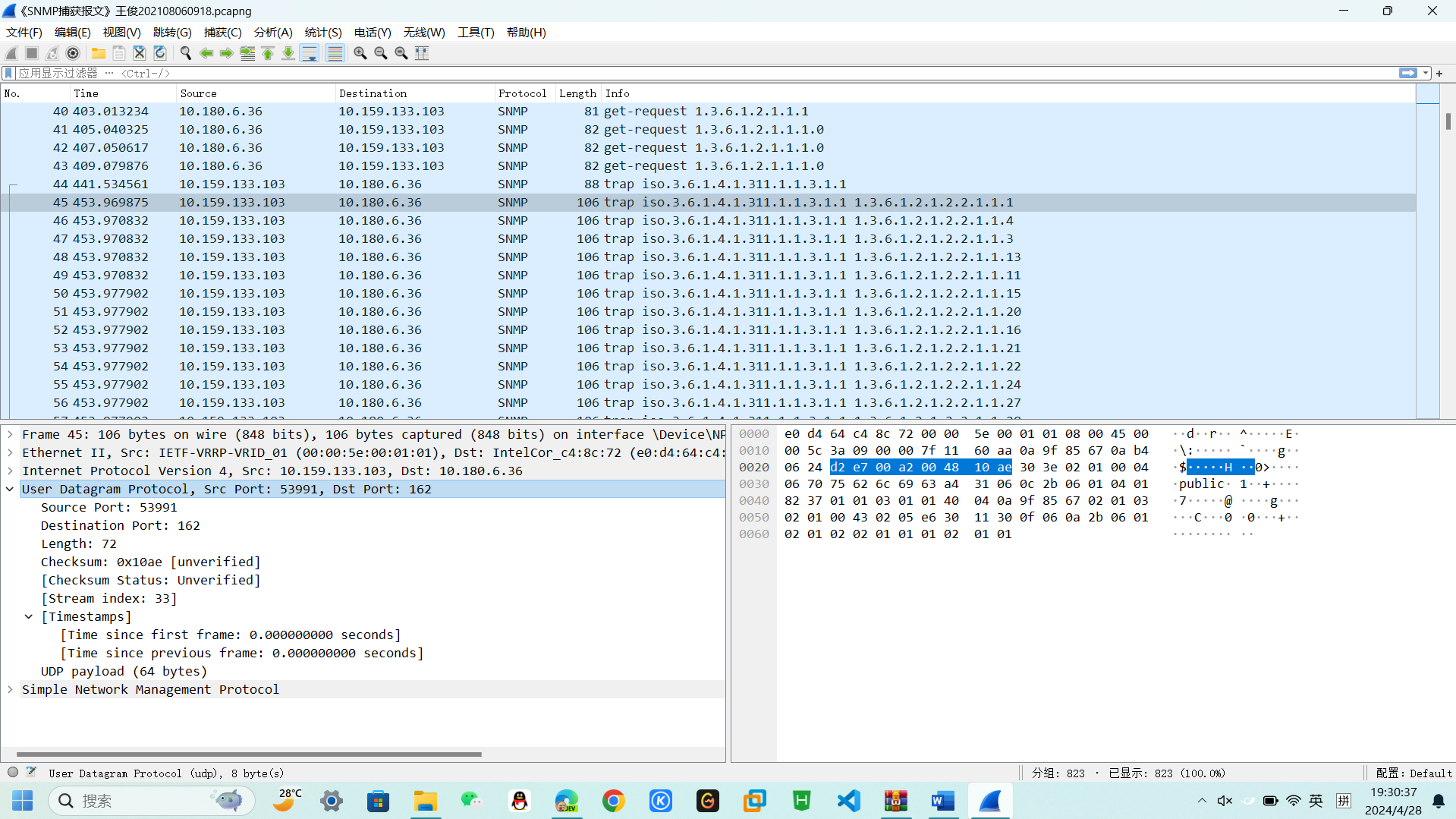


SNMP:

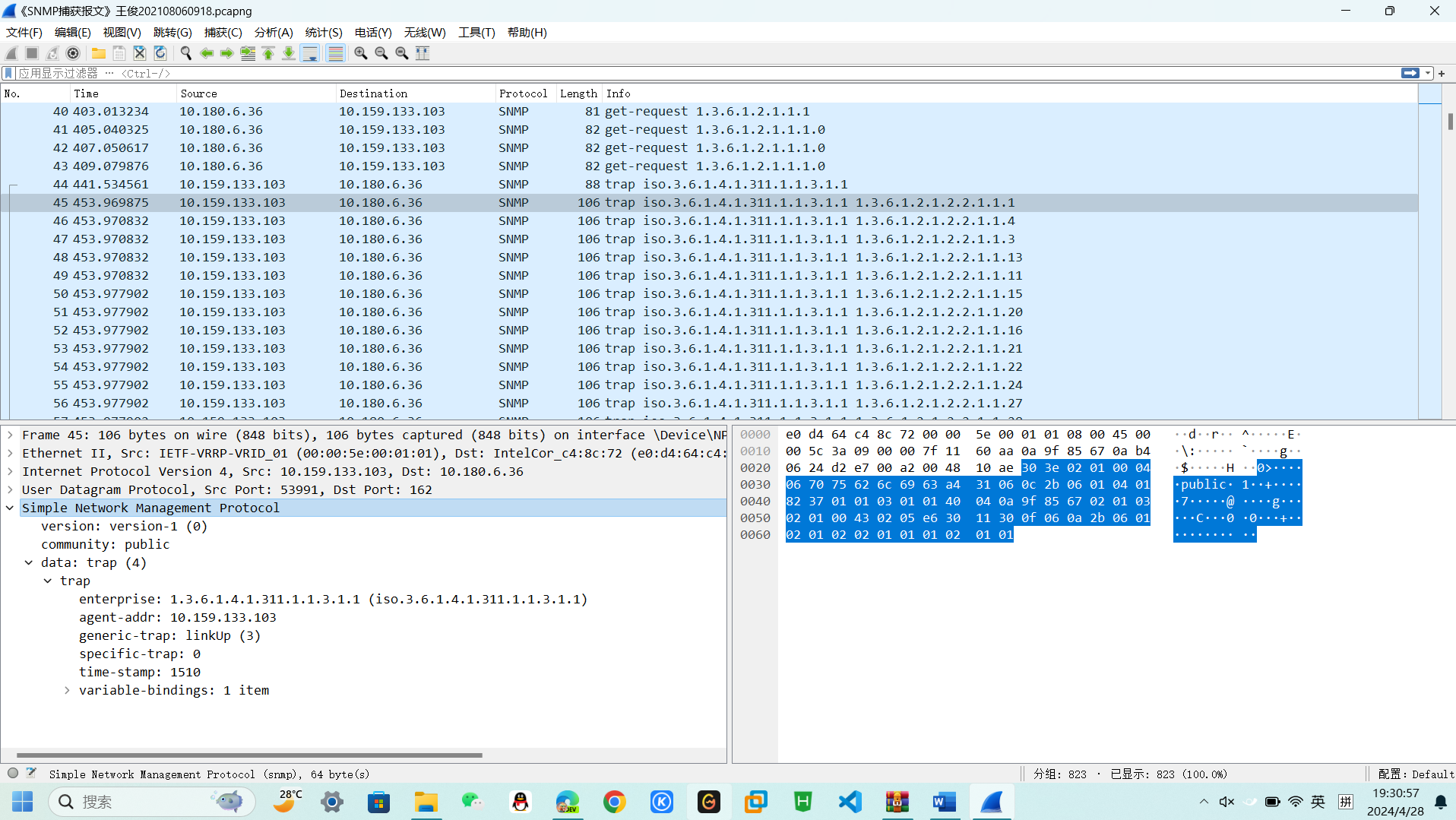


陷入报文：

UDP:



SNMP:



UDP头部：

UDP头部是所有基于UDP协议的数据包所共有的，对于SNMP的陷入报文、请求报文和应答报文来说也不例外。

UDP头部主要包括源端口号、目标端口号、包长度和校验和这几个字段。

源端口号：对于SNMP来说，源端口号标识了发送报文的设备或应用程序的端口。管理站通常使用UDP端口162发送请求报文，而代理则使用UDP端口161发送陷入报文和应答报文。

目标端口号：目标端口号标识了接收报文的设备或应用程序的端口。对于请求报文，目标端口是代理的161端口；对于陷入报文和应答报文，目标端口则是管理站的162端口。

包长度：包长度字段包含了UDP头部和数据的总长度。

校验和：校验和用于确保数据的完整性，它是对UDP头部和数据进行计算得出的。

SNMP报文头部：

SNMP报文头部是SNMP协议特有的，它包含了SNMP报文的关键信息。

报文头部：这部分包括版本号、社区名（或用户名，对于SNMPv3）和报文类型（如Get、Set、Trap等）。版本号指示了使用的SNMP协议版本，社区名用于身份验证和访问控制，报文类型则指明了报文的具体类型。

PDU头部：对于请求和响应报文，PDU头部包含了操作类型（如Get、Set等）、请求ID、错误状态和错误索引等字段。这些字段提供了关于SNMP操作的具体信息。对于陷入报文，PDU头部则包含了陷入类型、企业号、代理地址和通用陷阱信息等。

区别：

1. UDP头部：

源端口号与目标端口号：请求报文通常由管理站通过162端口发送，而陷入报文和应答报文则由代理分别通过161和162端口发送。

包长度：由于报文的数据部分长度不同（例如，陷入报文可能包含更详细的陷阱信息），因此包长度也会有所不同。

校验和：每个报文都有自己的校验和，用于确保数据完整性。

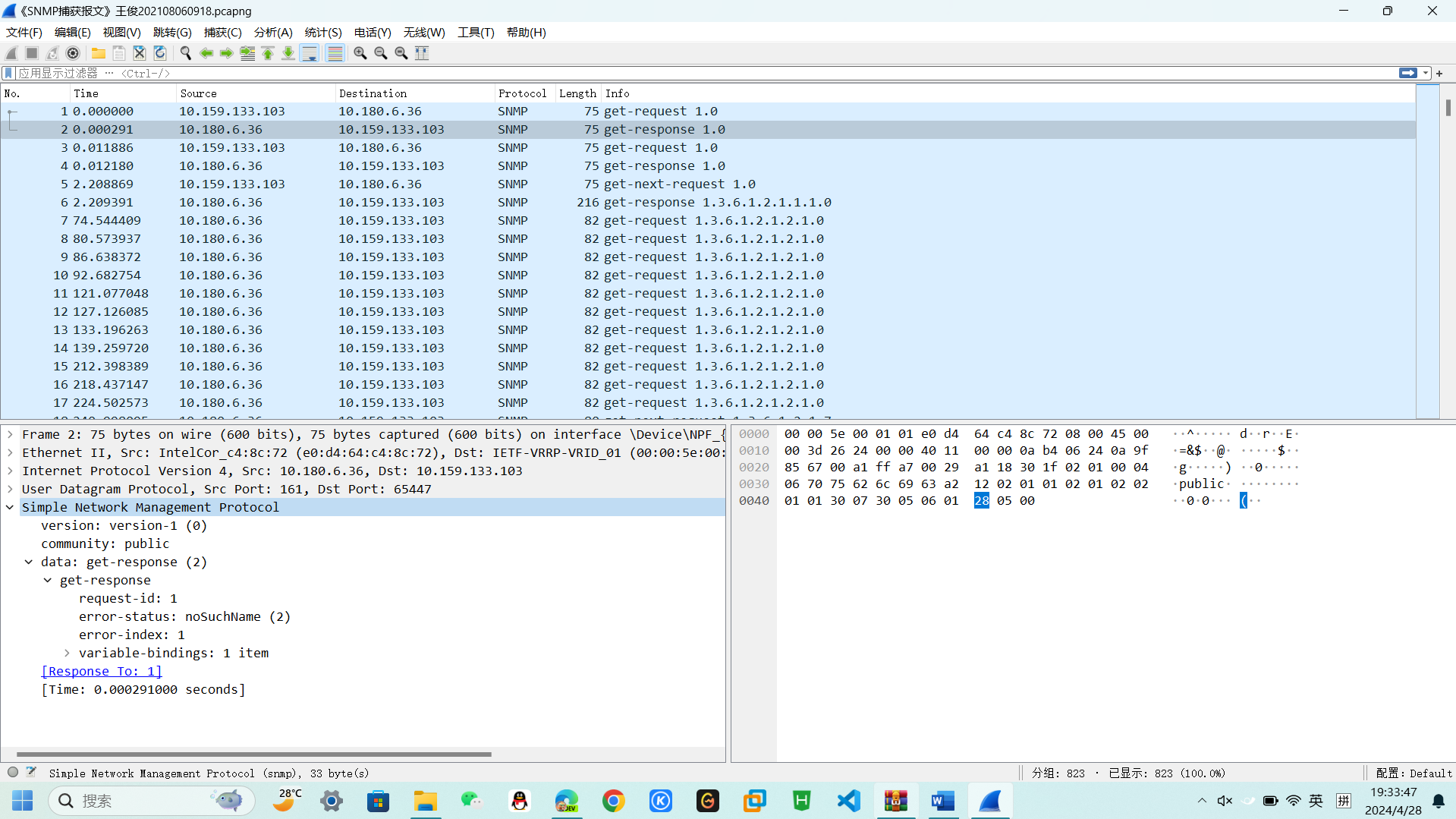
2. SNMP报文头部：

报文类型：请求报文可以是Get、Set等类型，应答报文则是对这些请求的响应；而陷入报文则是代理主动发送的，用于报告事件或异常。

PDU头部内容：对于请求和响应报文，PDU头部包含与特定操作相关的信息；对于陷入报文，它包含与陷阱事件相关的信息。

2、选取一条SNMP应答报文和一条陷入报文进行解析，分析SNMP报文和PDU各段值及其含义。

应答报文：



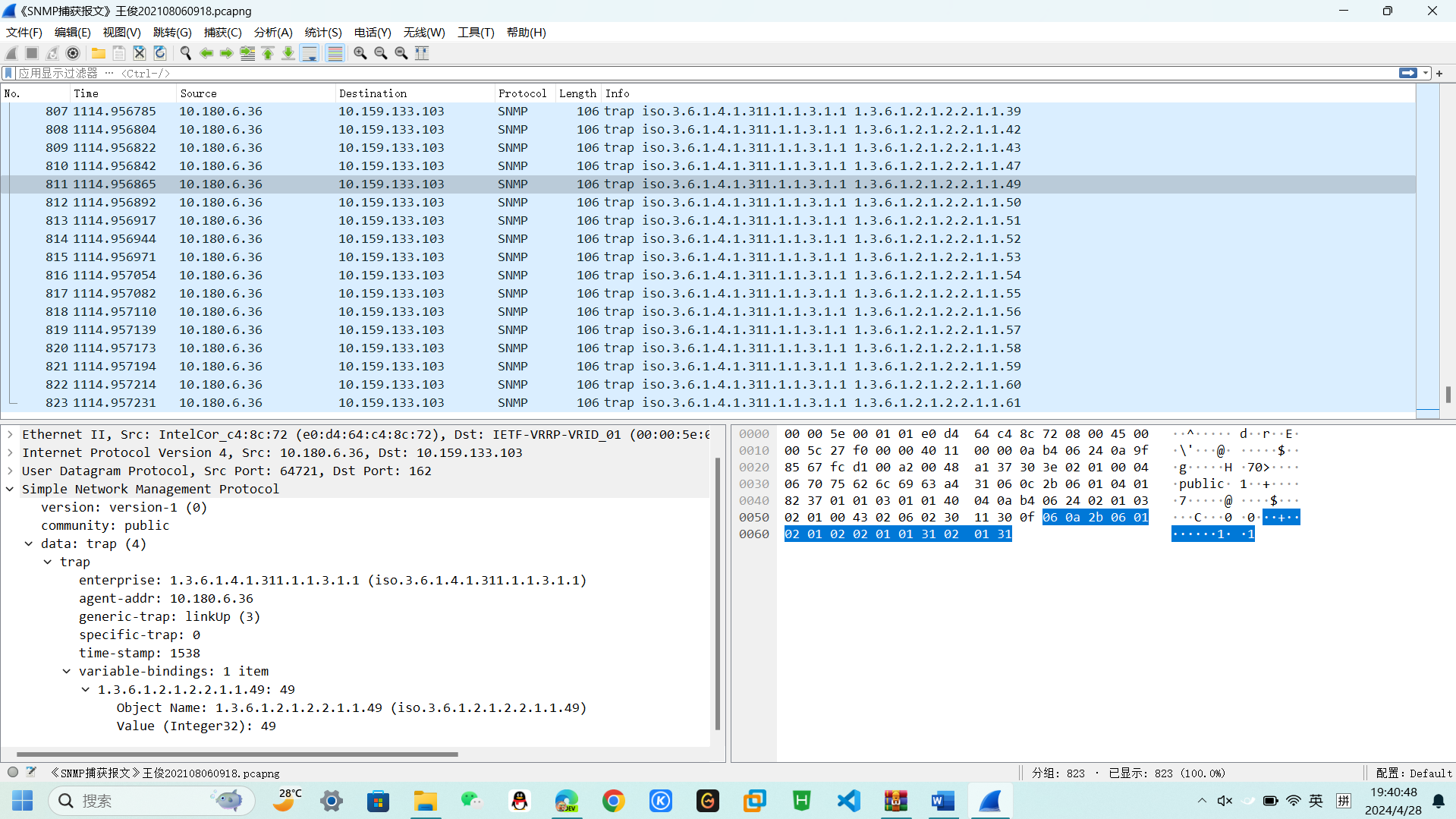
version:version-1(0) 代表SNMP版本为1

community:public 代表社区名称为public

data:get-response(2) 代表SNMP响应报文

get-response携带了所请求的信息或对象的值

Trap报文：



version:version-1(0) 代表SNMP版本为1

vommunity:public 代表社区名称为public

data:trap(4) 代表SNMP陷入报文，Trap PDU的标识符

**enterprise: 1.3.6.1.4.1.311.1.1.3.1.1**：这是企业私有OID（对象标识符）

agent-addr: 10.180.6.36：代理地址，即发送此Trap消息的网络设备的IP地址。

generic-trap: linkup (3)：通用Trap类型代码，这里是3，表示“linkup”，即某个网络链接已建立或恢复。

specific-trap: 0：特定Trap代码，对于通用Trap类型的一个补充说明.

time-stamp: 1538：时间戳，记录了Trap报文发送的时间。

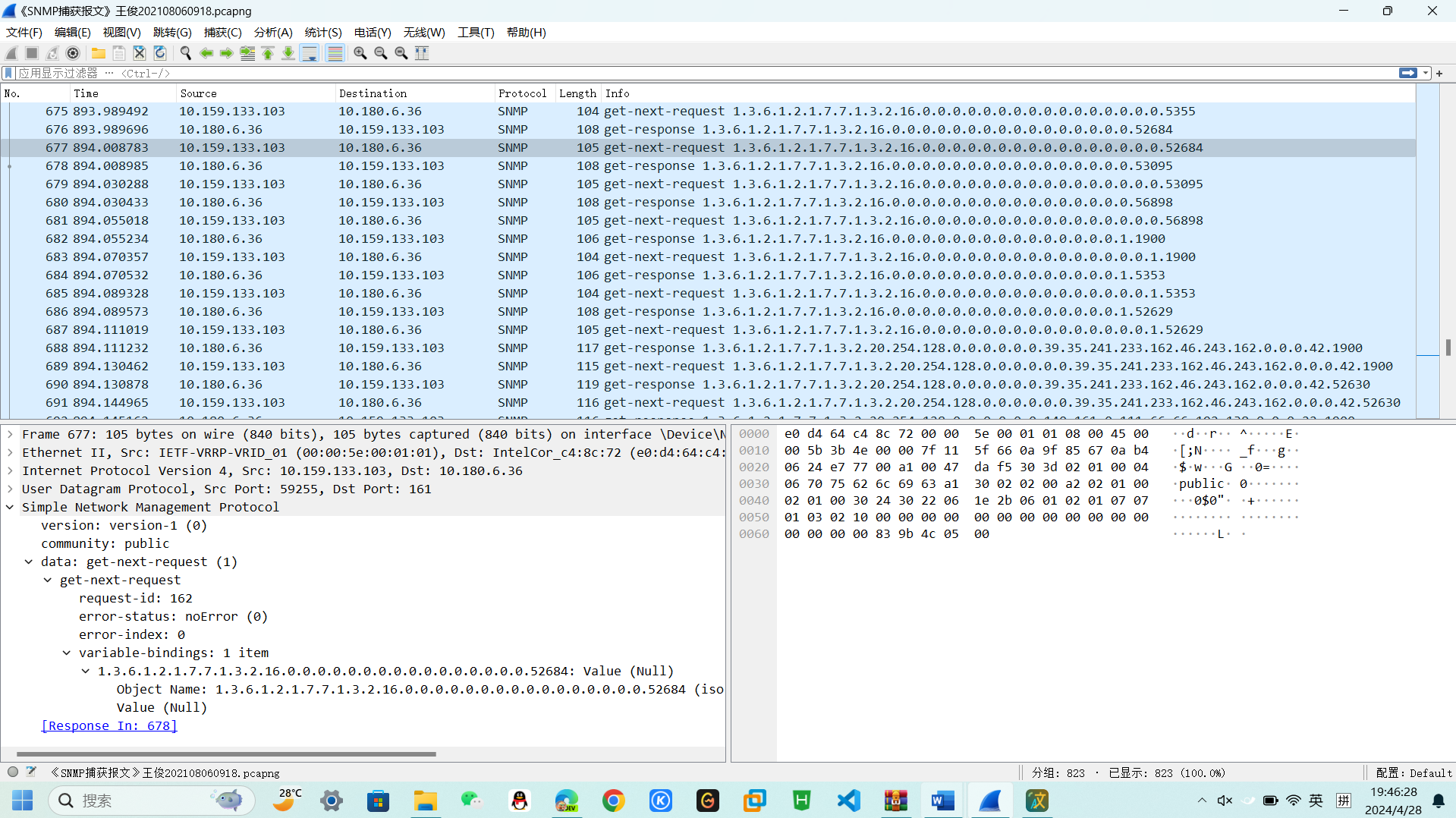
variable-bindings: 1 item：表示接下来有1个变量绑定信息。

1.3.6.1.2.1.2.2.1.1.49:49：这是变量绑定的详细信息，包括：

Object Name: 1.3.6.1.2.1.2.2.1.1.49，这是变量的OID。

Value (Integer32): 49，这是该OID对应的值，一个整数值49。

3、选取一协议数据单元（PDU）进行解析，写出各段数据类型、长度和值。（参见教材P25表2.1和P31 基本编码规则），并使用ASN.1描述所选择的PDU。



**SNMP Version and Community：**

Version: version-1(0) 表示这是SNMPv1协议版本。

Community: public 是团体名，

**PDU Type：**

Type: get-next-request (1)，表示这是一个GetNextRequest类型的PDU，用于请求获取MIB中下一个对象的值。

**Request Details：**

Request ID: 162，这是一个唯一的标识符，用于匹配请求和响应。

Error Status: noError (0)，表明请求过程中没有发生错误。

Error Index: 0，表示没有特定的错误索引。

**Variable Bindings：**

Bindings Count: 1 item，表示有1个变量绑定信息。

Variable Binding:

OID: 1.3.6.1.2.1.7.7.1.3.2.16.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.52684，这是请求中指定的下一个对象的OID。

Value: Value(Null)，说明在当前的MIB中，紧接着上一个查询对象的下一个对象不存在或没有值。

**ASN.1:**

SNMPv1-PDU ::= SEQUENCE {

version INTEGER,

community OCTET STRING,

pduType INTEGER,

requestID INTEGER,

errorStatus INTEGER,

errorIndex INTEGER,

variableBindings SEQUENCE OF VariableBinding

}

VariableBinding ::= SEQUENCE {

name OBJECT IDENTIFIER,

value ANY

}

SNMPv1-PDU WJPDU{

version 0,

community "public",

pduType 1,

requestID 162,

errorStatus 0,

errorIndex 0,

variableBindings {

VariableBinding {

name "1.3.6.1.2.1.7.7.1.3.2.16.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.52684",

value NULL

}

}

}

六、实验总结

通过实际操作SNMP协议，我更加深入地理解了课本上的理论知识，包括SNMP的基本工作流程、MIB的结构和OID的概念。

在实验过程中，我也遇到了以下问题：

1. **报文捕获问题**：使用数据包捕获工具（如Wireshark或Sniffer）时，可能无法捕获到SNMP报文。

**解决方案**: 确保网络接口正确选择，且处于混杂模式（如果需要），同时确认SNMP流量确实经过所监控的网络接口。检查防火墙和ACL设置，确保它们没有阻止SNMP流量。

1. **SNMP协议理解不足**：对SNMP协议的工作流程、报文类型（GET, GETNEXT, SET, TRAP等）和响应代码不熟悉。

**解决方案**: 加强理论学习，查阅SNMP协议标准文档（如RFCs），并通过实例分析加深理解。利用在线资源、教程和书籍来补充知识。

1. **MIB解析困难**：难以理解MIB-2树的结构，或是在实际报文中对应MIB对象。

**解决方案**: 使用MIB浏览器工具，如MIB Browser。

1. **配置错误**： SNMP代理或管理站配置不当，导致请求失败或未按预期响应。

**解决方案**: 仔细检查SNMP配置，包括团体名（community string）、访问控制列表（ACL）、版本选择（v1, v2c, v3）等，并确保两端配置一致。使用SNMP测试工具验证配置是否生效。