锐捷认证过程分析与第三方锐捷认证客户端的设计与实现

[摘要] 本文分析了锐捷认证过程，解释了锐捷认证过程中申请者（Supplicant）与验证者（Authenticator）往来的数据帧的格式和含义，以及锐捷加密认证数据的核心算法。本文提出了一套第三方锐捷认证客户端在Linux平台上实现的方法。以此方法为基础，实现了能够运行于 Linux 平台上的第三方锐捷认证客户端 jmuSupplicant。jmuSupplicant 拥有集美大学锐捷官方认证客户端的主要功能(上网认证，保持在线)。jmuSupplicant 在实现上网认证、保持在线这两个主要基础功能之上，结合集美大学锐捷认证漏洞，实现了普通用户在集美大学夜晚断网期间认证并保持在线的功能。本文中论述的所有理论背景均建立在集美大学锐捷认证环境之上，不涉及其他任何锐捷认证环境。

[关键词] 锐捷认证　第三方锐捷认证　Linux　断网认证

Ruijie Authentication Process Analysis and Design and Implementation of A Third-party Ruijie Authentication Client

[Abstruct] This paper analyzes the Ruijie authentication process and explains the format and meaning of the data frames sent by the supplicant and the authenticator during the Ruijie authentication process, and the core algorithms of Ruijie encryption authentication data. This paper presents a method about how to implement a third-party Ruijie authentication client on a Linux platform. Based on this method, the third-party Ruijie authentication client jmuSupplicant, which can run on the Linux platform, has the main functions of the Jimei University Ruijie official authentication client (network access authentication, keeping online). JmuSupplicant combines the two basic functions of network access authentication and keeping online. In conjunction with the Ruijie vulnerabilities of Jimei University, jmuSupplicant achieves the function of network access authentication and keeping online during the network disconnection at Jimei University at night. All theoretical backgrounds discussed in this article are based on the Ruijie certification environment at Jimei University and do not involve any other Ruijie certification environment.

[Keywords]

Ruijie authentication Third-party Ruijie certification Linux non-learning time authentication

目录

第 1 章 序言

在中国大陆，113 所大学里的校园网用户需要使用锐捷官方认证客户端认证[1]成功后，继而在互联网中畅游。在这 113 所需要锐捷认证的大学中，有一些大学提供的 Linux 版本锐捷官方客户端不可用，或是仅提供 windows 版本的锐捷官方客户端。这带给操作系统为 Linux 的用户无法接入校园网的困扰。一个能稳定运行于 Linux 平台上的第三方锐捷认证客户端，成为部分大学用户的迫切需求。令人愉快的是，锐捷官方客户端向验证者认证的过程不是一个黑盒，我们已经拥有一定量的资料和工具对锐捷官方客户端认证过程中的部分细节进行分析。

锐捷认证遵循 802.1X 协议[2]。802.1X 协议为 IEEE 802 媒体提供“网络端口认证”，这些媒体包括以太网（Ethernet），令牌环网（Token Ring）和 802.11 无线区域网络（802.11 wireless LAN）。实现 802.1X 协议的认证系统拥有 3 个实体：申请者（Supplicant）[2]、验证者（Authenticator）[2]、验证服务器（Authentication Server）[2]。802.1X 协议在网络端口处对用户设备进行控制，验证者对其所链接的申请者进行认证，验证服务器对其所链接的验证者进行认证。以上两组认证均成功，才能算申请者认证成功，此后用户设备即可访问互联网或其他区域网络（LAN）中的资源，如果申请者认证失败，则阻挡用户设备访问这些资源。

802.1X 协议认证系统架构如下图所示。

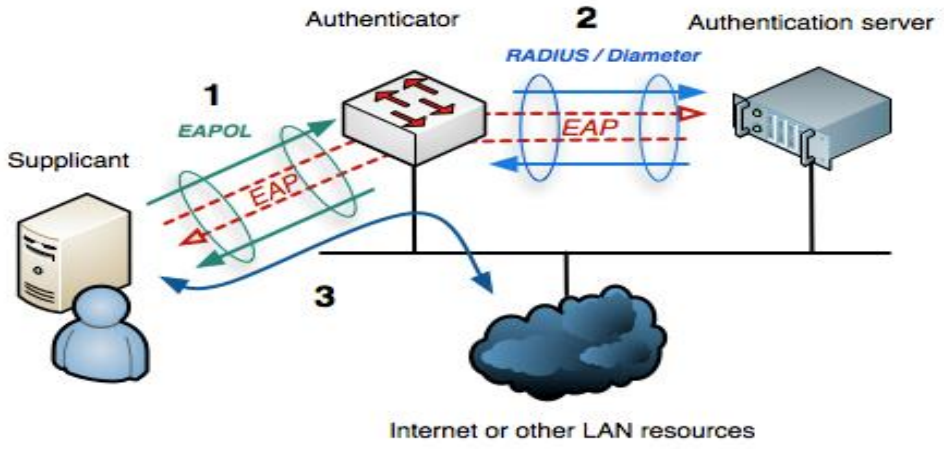


图 1- 1 802.1X 协议认证系统架构

申请者，它只和验证者进行通信，验证者和验证服务器之间的通讯对申请者来说是透明的。最简单的情况下，在一次成功的认证过程中，验证者会两次请求申请者发送指定的数据帧（data frame）给验证者，这两次向申请者请求的数据帧是不同的。申请者收到验证者的请求后在本地构造对应的数据帧返回给验证者。如果每一次申请者返回的数据帧中包含的数据，经验证服务器验证后都是正确的，那么申请者就能够认证成功。

锐捷官方认证客户端，在锐捷认证的过程中扮演的角色即申请者。实现一个第三方锐捷认证客户端，就是实现一个申请者。第三方锐捷认证客户端收到验证者的请求后，在本地构造对应的数据帧返回给验证者。如果第三方锐捷认证客户端构造的数据帧能够和锐捷官方认证客户端构造的一模一样（简单起见，实际只需关键数据段一模一样），那么它就能够替代锐捷官方客户端进行认证了。

虽然锐捷官方认证客户端是一个黑盒，我们无法知晓程序内部是如何构造数据帧的，但是我们可以从锐捷官方认证客户端的外部行为来推测出黑盒的运行逻辑。开源网络协议分析软件—wireshark[3]能够捕获锐捷官方认证客户端在认证过程中同验证者往来的数据帧，并且对每一个数据帧的格式类型，关键数据段的含义均有解释（精确到字节）。不仅能看到图 1-1 中的 EAP/EAPOL 协议[4]数据，还能看到 Ethernet 协议头数据，锐捷认证时需要的用户名，申请者 ip，申请者 netmask 等等。这些数据有的是明文，有的被加密。通过分析捕获到的数据帧的结构和含义，反向推测出锐捷官方认证客户端构造数据帧的逻辑，从而实现第三方锐捷认证客户端。

在这篇论文中，第二章介绍了本文的相关研究工作“MentoHUST[5]”项目和“hyrathb/mentohust[6]”项目。第三章介绍了锐捷官方认证客户端和验证者之间往来数据帧的结构和含义。第四章介绍了锐捷官方认证客户端加密认证数据时使用的算法，包括“计算 IP 地址”，“计算 EAP-MD5-Value”等。第五章介绍了第三方锐捷认证客户端 jmuSupplicant 的实现过程。第六章是对本文的总结和工作展望。

以下内容中，“锐捷官方认证客户端”简称“锐捷客户端”，“第三方锐捷认证客户端 jmuSupplicant”简称“jmuSupplicant”。

第 2 章 相关研究工作

2.1 MentoHUST 项目

在第三方锐捷认证客户端中，以 MentoHUST 项目最为著名。MentoHUST 是一个支持 Windows、Linux、Mac OS 下锐捷认证的程序（附带支持赛尔认证)。Windows 版 MentoHUST 支持 Windows 所有主流版本，与锐捷官方认证程序相比最大的优势是内存占用低。Linux 版 MentoHUST 是一个在 Linux 下与锐捷兼容性很好的认证客户端，方便使用 Linux 和锐捷的同学使用校园网。在成员 kkHAIKE 的努力下，MentoHUST 现已支持锐捷的 V3 客户端校验算法。Mac OS 版 MentoHUST 是后来对 Linux 版 MentoHUST 在 Mac OS 上的编译[5]。MentoHUST 算是最初代的第三方锐捷认证客户端，由于原作者早已毕业，项目停止维护很久了。之后，锐捷公司升级了锐捷认证算法，由 V3 客户端校验算法升级到了 V4 客户端校验算法，MentoHUST 无法再认证成功。

2.2 hyrathb/mentohust 项目

hyrathb/mentohust 项目原本旨在为华中科技大学的用户提供 Linux 平台上的锐捷认证功能，但中国大陆许多的锐捷认证环境与华中科技大学的锐捷认证环境兼容，因此 hyrathb/mentohust 同样适用于除华中科技大学外的许多大学。hyrathb/mentohust 是目前流传最广的可用的第三方锐捷认证客户端版本。hyrathb/mentohust 项目最大贡献是解决了锐捷认证 V4 客户端校验算法。本文将该算法计算结果称为 V4 加密值[6]。V4 客户端校验算法会在本文第 4 章 4.3 节中介绍。在此对 hyrathb/mentohust 项目作者表示感谢。

第 3 章 分析锐捷认证数据帧

如图 1-1 所示，在申请者与验证者之间，EAP 协议(Extensible Authentication Protocol）[4]承载着认证信息，保证认证的安全性。EAPOL 协议(Extensible Authentication Protocol Over Lan Protocol）位于数据链路层（Data-link Layer），其功能是封装并传递 EAP 协议数据，使得申请者和验证者之间能够在区域网络（包括以太网和无线区域网络）中通信。RFC 3748（Extensible Authentication Protocol (EAP)）详细介绍了 EAP 协议的工作逻辑、格式、请求/回复类型等 EAP 协议细节。

锐捷认证过程中，申请者一共可能会从验证者接收到 4 种数据帧。分别是 EAP-Request-Identity ，EAP-Request-MD5-Challenge，EAP-Success 和 EAP-Failure[4]。一共可能会发送给验证者 4 种数据帧，分别是 EAPOL-Start，EAP-Response-Identity，EAP-Response-MD5-Challenge， EAPOL-Logoff。其中，EAP-Failure 数据帧和 EAPOL-Logoff 数据帧较为特殊，这两种数据帧在一次成功的锐捷认证过程中是不会出现的。EAP-Failure 数据帧用于申请者认证失败的情况，当申请者发送给验证者的数据帧不正确时，验证者立刻返回 EAP-Failure 数据帧告知申请者此次认证失败。EAPOL-Logoff 数据帧用于已经认证成功的申请者主动下线的情况，当验证者收到 EAPOL-Logoff 数据帧时，验证者会关闭链接申请者的端口，使主动要求下线的申请者无法再访问互联网或其他区域网络中的资源[1]。一次成功的锐捷认证过程，数据帧在申请者和验证者之间往来的顺序如下图所示。

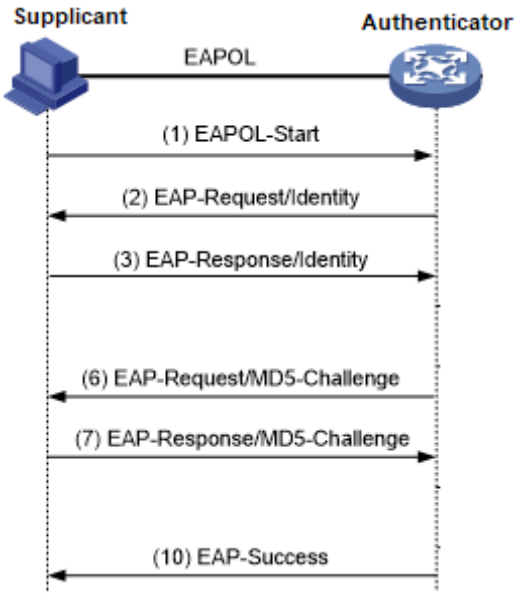


图 3- 1 锐捷认证过程数据帧顺序

下面按照图 3-1 中数据帧在申请者和验证者之间往来的顺序[7]，详解 EAPOL-Start、EAP-Request-Identity、EAP-Response-Identity、EAP-Request-MD5-Challenge、EAP-Response-MD5-Challenge，5 种锐捷认证数据帧。

3.1 EAPOL-Start 数据帧

申请者主动对外发送 EAPOL-Start 数据帧是锐捷认证的第 1 步。使用 wireshark 软件捕获锐捷认证过程中 EAPOL-Start 数据帧元数据如下图所示。

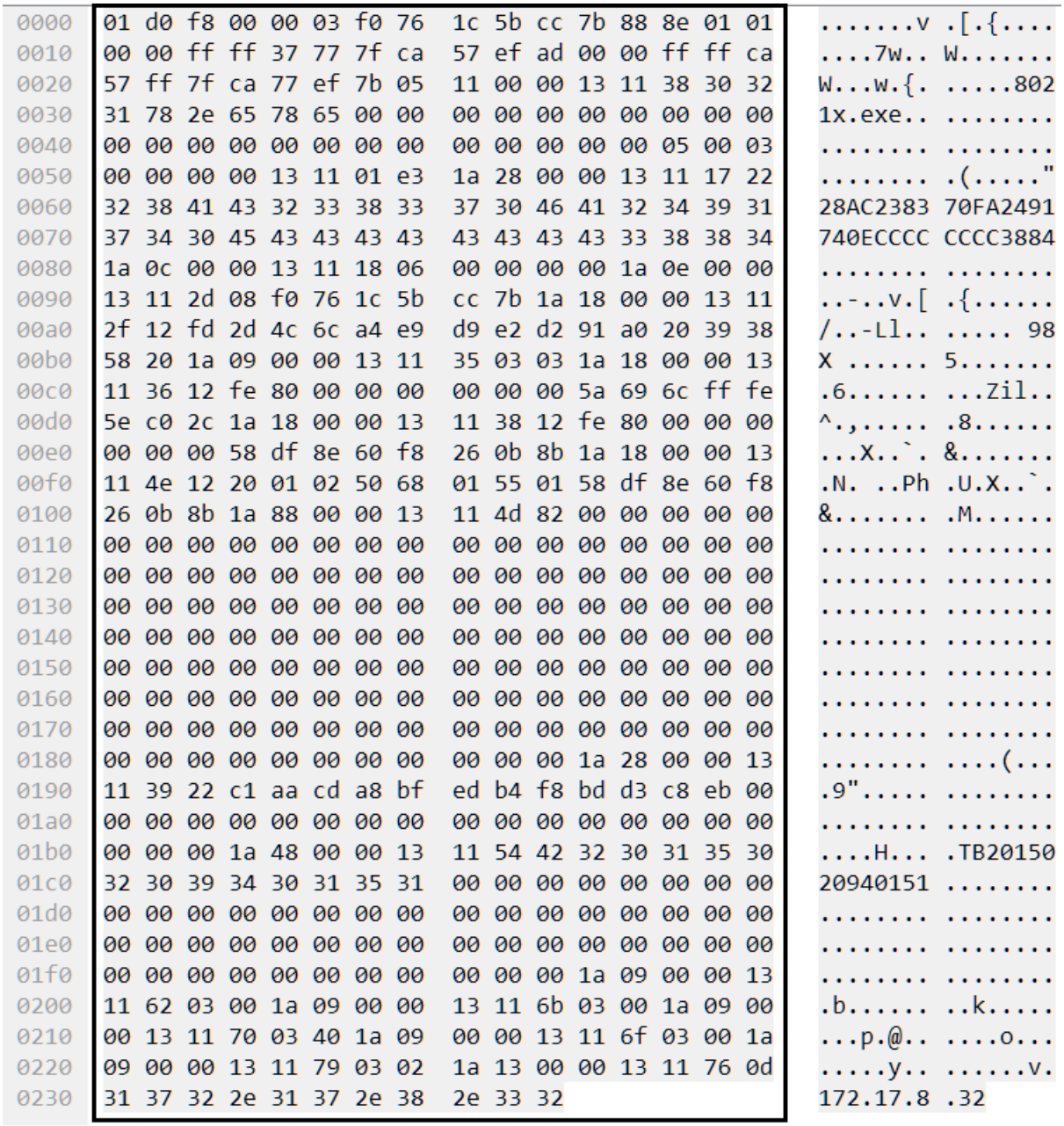


图 3- 2 EAPOL-Start 数据帧

图 3-2 黑框中的数据为 EAPOL-Start 数据帧的元数据，黑框左侧标注了每行第一个字节数据的位置，黑框右侧是 wireshark 软件自动对黑框中元数据的翻译，便于使用者更直观地理解。

EAPOL-Start 数据帧格式如下图所示。

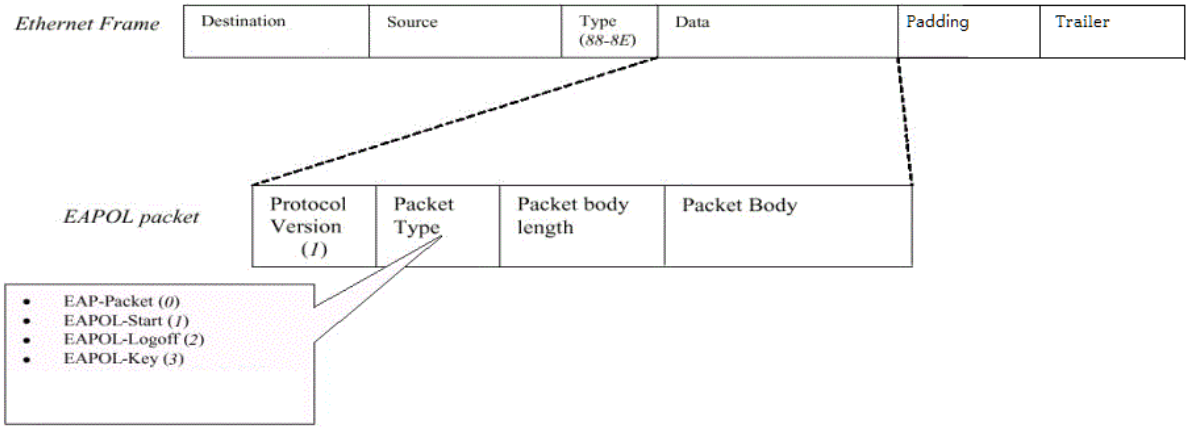


图 3- 3 EAPOL-START 数据帧格式

下面将按照图 3-3 中标注的 EAPOL-START 数据帧格式，对照图 3-2 中的 EAPOL-START 数据帧元数据，按顺序解释锐捷认证时需要用到的各段元数据的含义。

图 3-3 中 Destination 的范围对应图 3-2 中的位置在 0x0000-0x0005 之间，固定值为 0180c2000003。EAPOL-Start 数据帧用于申请者主动向验证者请求认证，对申请者来说，此时验证者位于数据链路层的 MAC 地址（Media Access Control Address）是未知的。为了让申请者找到未知的验证者，在 EAPOL-Start 数据帧中，以太网协议头（Ethernet header）中目的地址（Destination Address）固定为 802.1X 协议的端口存取实体地址（Port Access Entity Address），01:80:c2:00:00:03。

图 3-3 中 Source 的范围对应图 3-2 中的位置在 0x0006-0x000B 之间，值为 f0761c5bcc7b。Source 是申请者网络设备的 MAC 地址，是由申请者的设备决定的。这个 MAC 地址为后序验证者发送数据帧给申请者提供了目的 MAC 地址。

图 3-3 中 Type 的范围对应图 3-2 中的位置在 0x000C-0x000D 之间，固定值为 888e。802.1X 协议的协议号为两个字节的 888e（0x888e）。当以太网帧的 Type 为 888e 时，表示此以太网帧中包含一个 EAPOL 数据帧。

图 3-3 中 Protocol Version 的范围对应图 3-2 中的位置在 0x000E，固定值为 01。Protocol Version 代表 802.1X 认证版本为 802.1X-2001。

图 3-3 中 Packet Type 的范围对应图 3-2 中的位置在 0x000F，固定值为 01。Packet Type 代表当前 EAPOL 数据帧的种类为 EAPOL-Start。申请者用此值告诉验证者此刻需要初始化认证过程。初始化认证过程这一步骤对验证者来说是非常必要的，序言中提到 802.1X 协议为 IEEE 802 媒体提供“网络端口认证”，初始化认证过程代表原本对申请者“关闭”的网络端口此时“开启”了。

图 3-3 中 Packet body length 的范围对应图 3-2 中的位置在 0x0010-0x0011 之间，固定值为 0000，Packet body length 表示在 EAPOL-Start 数据帧中封装的 EAP 数据的长度。为 0000 意味着 EAPOL-Start 数据帧中没有封装 EAP 数据。这是因为 EAPOL-Start 数据帧没有认证信息需要发送给验证者，EAPOL-Start 数据帧的作用就是简单地通知验证者此刻有一个申请者要与验证者进行通讯，验证者需要初始化认证过程。

图 3-2 中 0x0017-0x001a 数据段是申请者的 IP 地址，值为 ca57efad。IP 地址的值由申请者的 IP 地址决定，且被锐捷算法加密过，不是明文传输。具体的算法逻辑见第 4 章 4.1 节。

图 3-2 中 0x001b-0x00e 数据段是申请者的子网掩码（Subnet Mask）地址，值为 0000ffff。子网掩码地址的值由申请者的子网掩码地址决定，且被锐捷算法加密过，不是明文传输。具体的算法逻辑见第 4 章 4.1 节。在多次修改子网掩码地址后分析得知，锐捷认证时不会校验子网掩码地址的正确性。

图 3-2 中 0x001f-0x0023 数据段是申请者的网关（Gateway）地址，值为 ca57ff7f。网关地址的值由申请者的网关地址决定，且被锐捷算法加密过，不是明文传输。具体的算法逻辑见第 4 章 4.1 节。在多次修改网关地址后分析得知，锐捷认证时不会校验网关地址的正确性。

图 3-2 中 0x0193-0x019e 数据段是认证服务名，值为 c1aacda8bfedb4f8bdd3c8eb。这一段是汉字“联通宽带接入”转换为 GB2312 格式[8] **错误!未找到引用源。**后的值。集美大学校园网一共提供 4 种认证服务，另外三种分别为“教育网接入”，值为 bdccd3fdcdf8bdd3c8eb；移动宽带接入，值为 d2c6b6afbfedb4f8bdd3c8eb；电信宽带接入，值为 b5e7d0c5bfedb4f8bdd3c8eb。在多次修改服务名后分析得知，锐捷认证时不会校验 EAPOL-Start 数据帧中服务名的正确性。

图 3-2 中 0x0230-0x023a 数据段是申请者的域名系统（Domain Name System）地址，值为 3137322e31372e382e3332。从图 3-2 黑框右侧 wireshark 的翻译可知，此 DNS 地址为 172.17.8.32。域名系统的值由申请者的域名地址决定，明文传输。在多次修改域名地址后分析得知，锐捷认证时不会校验域名地址的正确性。

图 3-2 中其余未提到的数据段，目前暂未弄清楚它们的含义和作用。jmuSupplicant 构造 EAPOL-Start 数据帧时，保持这些数据段与锐捷客户端一致，不影响认证结果。

以上为对锐捷认证过程中 EAPOL-Start 数据帧的分析。

3.2 EAP-Request-Identity 数据帧

验证者向发送 EAPOL-Start 数据帧的申请者回应 EAP-Request-Identity 数据帧是锐捷认证的第 2 步。使用 wireshark 软件捕获锐捷认证过程中 EAP-Request-Identity 数据帧元数据如下图所示。

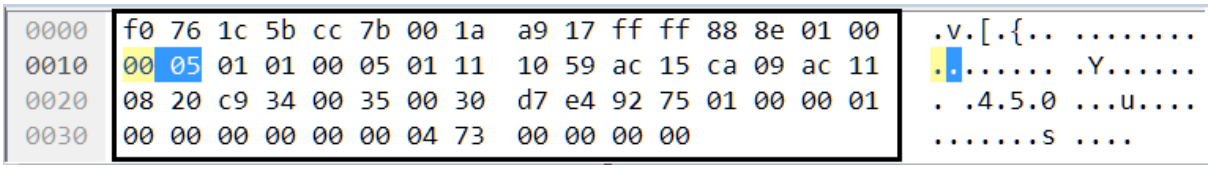


图 3- 4 EAP-Request-Identity 数据帧

图 3-4 黑框中的数据为 EAP-Request-Identity 数据帧的元数据，黑框左侧标注了每行第一个字节数据的位置，黑框右侧是 wireshark 软件自动对黑框中元数据的翻译，便于使用者更直观地理解。

EAP-Request-Identity 数据帧格式如下图所示。

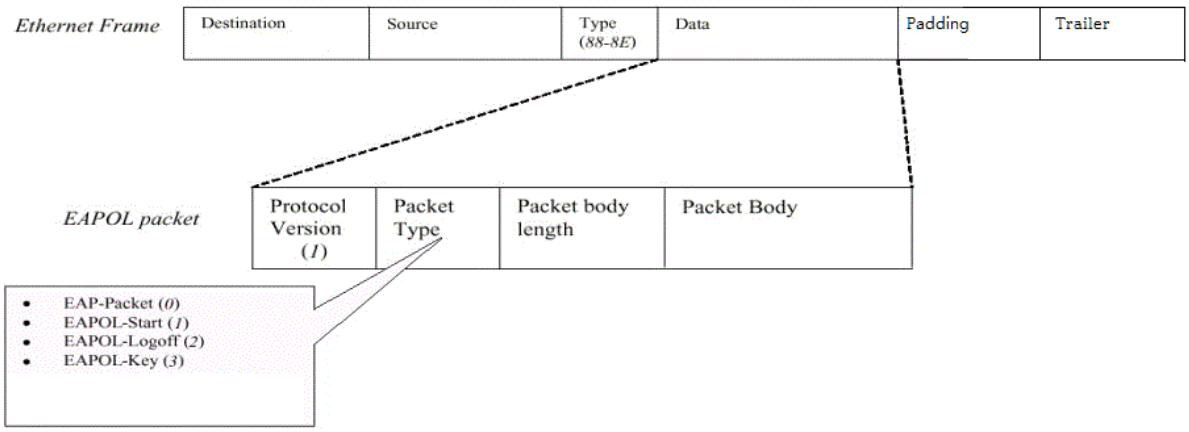


图 3- 5 EAP-Request-Identity 数据帧格式

下面将按照图 3-5 中标注的 EAP-Request-Identity 数据帧格式，对照图 3-4 中的 EAP-Request-Identity 数据帧元数据，按顺序解释锐捷认证时需要用到的各段元数据的含义。

图 3-5 中 Destination 的范围对应图 3-4 中的位置在 0x0000-0x0005 之间，值为 f0761c5bcc7b。Destination 为申请者网络设备的 MAC 地址。因为验证者收到了申请者发送的 EAPOL-Start 数据帧，所以验证者返回 EAP-Request-Identity 数据帧给申请者时，申请者位于数据链路层的 MAC 地址对验证者来说是已知的。

图 3-5 中 Source 的范围对应图 3-4 中的位置在 0x0006-0x000B 之间，值为 001aa917ffff。Source 是验证者网络设备的 MAC 地址，是由验证者的网络设备决定的。

图 3-5 中 Type 的范围对应图 3-4 中的位置在 0x000C-0x000D 之间，固定值为 888e。802.1X 协议的协议号为两个字节的 888e（0x888e）。当以太网帧的 Type 为 888e 时，表示此以太网帧中包含一个 EAPOL 数据帧。

图 3-3 中 Protocol Version 的范围对应图 3-2 中的位置在 0x000E，固定值为 01。Protocol Version 代表 802.1X 认证版本为 802.1X-2001。

图 3-5 中 Packet Type 的范围对应图 3-4 中的位置在 0x000F，固定值为 00。Packet Type 代表当前 EAPOL 的种类为 EAP 数据包。EAP 被认为是一种特殊的 EAPOL，当 Packet Type 被设置为 00 时，EAP 数据会直接通过 EAPOL 层不做任何的处理，也就是说 EAP 数据剥离了 EAPOL 的封装。因此，这也是当前数据帧称作 EAP-Request-Identity 的原因。

图 3-5 中 Packet body length 的范围对应图 3-4 中的位置在 0x0010-0x0011 之间，固定值为 0005，Packet body length 表示在 EAP-Request-Identity 数据帧中封装的 EAP 数据包的长度。为 0005 意味着封装了 5 个字节的 EAP 数据。

图 3-4 中 0x0012-0x0006 之间的 16 进制数据段为 EAP-Request-Identity 数据帧中封装的 EAP 数据包。EAP 数据包的格式如下。

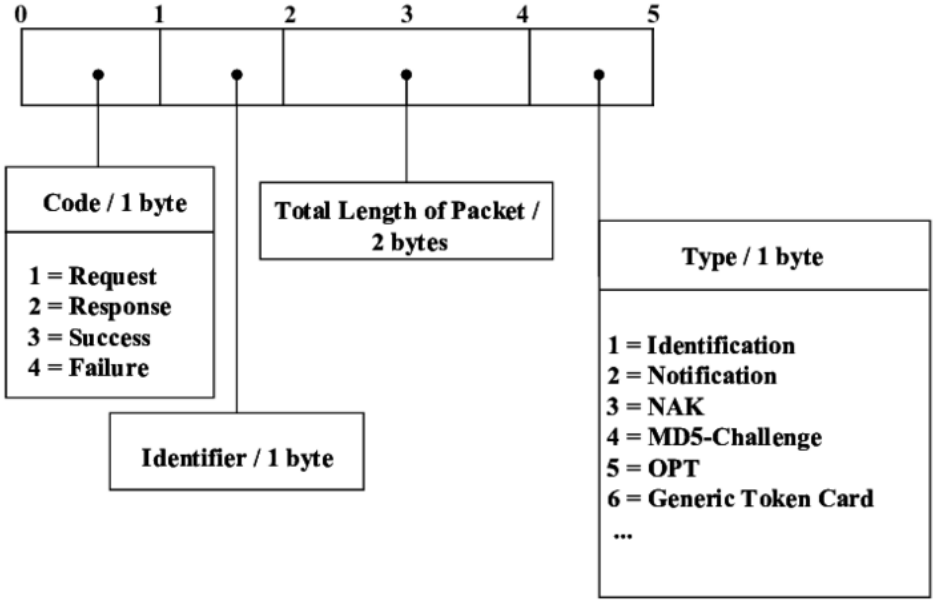


图 3- 6 EAP-Request-Identity 中 EAP 数据包格式

图 3-6 中 Code 的范围对应图 3-4 中的位置在 0x0012，固定值为 01。Code 定义了 EAP 数据包的类型。当 EAP 数据包的类型为 Request 时，值为 01。

图 3-6 中 Identifier 的范围对应图 3-4 中的位置在 0x0013，值为 01。若出现 Request 数据帧因超时发生重传时，重传后的 Identifier 要在重传前的 Identifier 值的基础上加 1。此时简化处理，忽略异常情况。

图 3-6 中 Total Length of Packet 的范围对应图 3-4 中的位置在 0x0014-0x0015 之间，固定值为 0005。Total Length of Packet 和图 3-5 中 Packet body length 表示的是同一个长度，即 EAP-Request-Identity 数据帧中封装的 EAP 数据包的长度。

图 3-6 中 Type 的范围对应图 3-4 中的位置在 0x0016，固定值为 01。Type 代表初始 EAP Request/Response 类型（Initial EAP Request/Response Types）。EAP-Request-Identity 类型为 Identity 时，因此值为 01。

图 3-4 中其余未提到的数据段，目前暂未弄清楚它们的含义和作用。

以上为对锐捷认证过程中 EAP-Request-Identity 数据帧的分析。

3.3 EAP-Response-Identity 数据帧

申请者向验证者发送 EAP-Reponse-Identity 数据帧是锐捷认证的第 3 步。使用 wireshark 软件捕获锐捷认证过程中 EAP-Response-Identity 数据帧元数据如下图所示。

```
0000  00 1a a9 17 ff ff f0 76  1c 5b cc 7b 88 8e 01 00   .......v .[.{....
0010  00 11 02 01 00 11 01 32  30 31 34 32 31 31 32 31   .......2 01421121
0020  30 35 39 ff ff 37 77 7f  ca 57 ef ad 00 00 ff ff   059..7w. .W......
0030  ca 57 ff 7f ca 77 ef 7b  05 11 00 00 13 11 38 30   .W...w.{ ......80
0040  32 31 78 2e 65 78 65 00  00 00 00 00 00 00 00 00   21x.exe. ........
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 05 00   ........ ........
0060  03 00 00 00 00 13 11 01  e3 1a 28 00 00 13 11 17   ........ ..(.....
0070  22 32 38 41 43 32 33 38  33 37 30 46 41 32 34 39   "28AC238 370FA249
0080  31 37 34 30 45 43 43 43  43 43 43 43 43 43 33 38 38   1740ECCC CCCCC388
0090  34 1a 0c 00 00 13 11 18  06 00 00 00 00 1a 0e 00   4....... ........
00a0  00 13 11 2d 08 f0 76 1c  5b cc 7b 1a 18 00 00 13   ...-..v. [.{.....
00b0  11 2f 12 fd 2d 4c 6c a4  e9 d9 e2 d2 91 a0 20 39   ./..-Ll. ...... 9
00c0  38 58 20 1a 09 00 00 13  11 35 03 03 1a 18 00 00   8X ..... .5......
00d0  13 11 36 12 fe 80 00 00  00 00 00 00 5a 69 6c ff   ..6..... ....Zil.
00e0  fe 5e c0 2c 1a 18 00 00  13 11 38 12 fe 80 00 00   .^.,.... ..8.....
00f0  00 00 00 00 58 df 8e 60  f8 26 0b 8b 1a 18 00 00   ....X..` .&......
0100  13 11 4e 12 20 01 02 50  68 01 55 01 58 df 8e 60   ..N. ..P h.U.X..`
0110  f8 26 0b 8b 1a 88 00 00  13 11 4d 82 34 32 39 31   .&...... ..M.4291
0120  39 64 30 36 62 32 31 36  33 62 34 37 35 32 63 62   9d06b216 3b4752cb
0130  30 63 36 61 64 30 34 32  66 32 38 62 61 30 32 63   0c6ad042 f28ba02c
0140  61 38 62 63 66 63 62 38  65 33 31 34 39 64 66 36   a8bcfcb8 e3149df6
0150  33 65 31 30 34 37 62 37  32 34 62 31 39 35 36 39   3e1047b7 24b19569
0160  63 66 33 63 33 37 32 65  35 39 33 39 66 39 65 38   cf3c372e 5939f9e8
0170  30 63 35 30 30 65 38 38  65 32 37 36 36 33 64 33   0c500e88 e27663d3
0180  39 64 63 32 30 37 65 33  33 39 65 33 61 30 63 35   9dc207e3 39e3a0c5
0190  63 63 61 33 65 36 65 31  39 61 65 30 1a 28 00 00   cca3e6e1 9ae0.(..
01a0  13 11 39 22 c1 aa cd a8  bf ed b4 f8 bd d3 c8 eb   ..9"....  ........
01b0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
01c0  00 00 00 00 1a 48 00 00  13 11 54 42 32 30 31 35   .....H.. ..TB2015
01d0  30 32 30 39 34 30 31 35  31 00 00 00 00 00 00 00   02094015 1.......
01e0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
01f0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0200  00 00 00 00 00 00 00 00  00 00 00 00 1a 09 00 00   ........ ........
0210  13 11 62 03 00 1a 09 00  00 13 11 6b 03 00 1a 09   ..b..... ...k....
0220  00 00 13 11 70 03 40 1a  09 00 00 13 11 6f 03 00   ....p.@. .....o..
0230  1a 09 00 00 13 11 79 03  02 1a 13 00 00 13 11 76   ......y. .......v
0240  0d 31 37 32 2e 31 37 2e  38 2e 33 32               .172.17. 8.32
```

图 3- 7 EAP-Response-Identity 数据帧

图 3-7 黑框中的数据为 EAP-Response-Identity 数据帧的元数据，黑框左侧标注了每行第一个字节数据的位置，黑框右侧是 wireshark 软件自动对黑框中元数据的翻译，便于使用者更直观地理解。

EAP-Response-Identity 数据帧格式如下图所示。

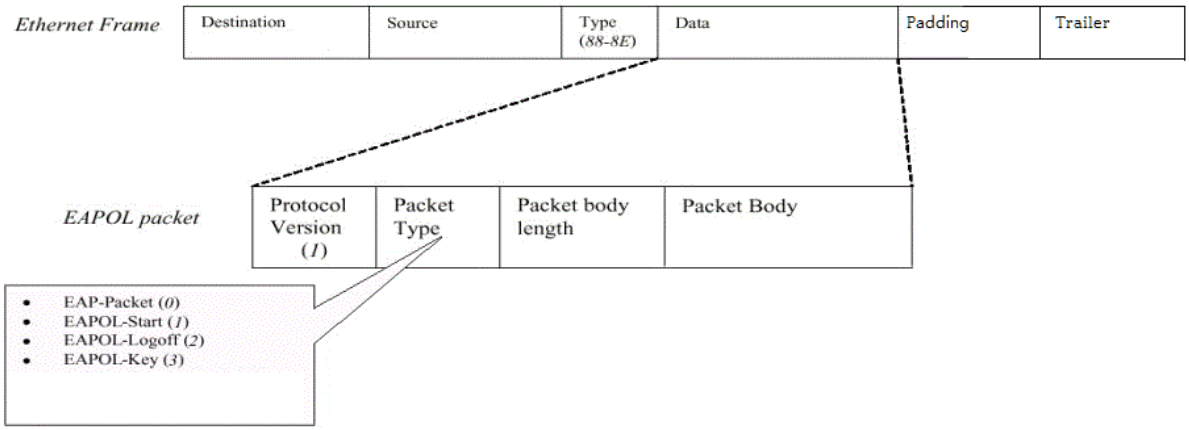


图 3- 8 EAP-Response-Identity 数据帧格式

下面将按照图 3-8 中标注的 EAP-Response-Identity 数据帧格式，对照图 3-7 中的 EAP-Response-Identity 数据帧元数据，按顺序解释锐捷认证时需要用到的各段元数据的含义。

图 3-3 中 0x0000-0x0005 数据段是申请者网络设备的 MAC 地址，值为 f0761c5bcc。此值为申请者的 MAC 地址。

图 3-3 中 0x0006-0x000c 数据段是验证者网络设备的 MAC 地址，值为 001aa917ffff。此值由申请者链接的验证者网络设备的 MAC 地址决定。

EAP-Request-Identity 数据帧是在申请者与验证者链接建立之后，由验证者发送给申请者的。

图 3-8 中 Destination 的范围对应图 3-7 中的位置在 0x0000-0x0005 之间，值为 001aa917ffff。Destination 是验证者网络设备的 MAC 地址。在申请者收到验证者发送的 EAP-Request-Identity 数据帧后，验证者位于数据链路层的 MAC 地址对申请者来说已知了，后续申请者发送认证数据帧的目的地址均为此地址，不再使用 EAPOL-Start 数据帧中的端口存取实体地址 01:80:c2:00:00:03。

图 3-8 中 Source 的范围对应图 3-7 中的位置在 0x0006-0x000B 之间，值为 f0761c5bcc7b。Source 是申请者网络设备的 MAC 地址。

图 3-8 中 Type 的范围对应图 3-7 中的位置在 0x000C-0x000D 之间，固定值为 888e。802.1X 协议的协议号为两个字节的 888e（0x888e）。当以太网帧的 Type 为 888e 时，表示此以太网帧中包含一个 EAPOL 数据帧。

图 3-8 中 Protocol Version 的范围对应图 3-7 中的位置在 0x000E，固定值为 01。Protocol Version 代表 802.1X 认证版本为 802.1X-2001。

图 3-8 中 Packet Type 的范围对应图 3-7 中的位置在 0x000F，固定值为 00。Packet Type 代表当前 EAPOL 的种类为 EAP 数据包。

图 3-8 中 Packet body length 的范围对应图 3-7 中的位置在 0x0010-0x0011 之间，固定值为 0011，Packet body length 表示在 EAP-Response-Identity 数据帧中封装了长度为 17 个字节的 EAP 数据包。

图 3-8 中 0x0012-0x0022 之间的 16 进制数据段为 EAP-Response-Identity 数据帧中封装的 EAP 数据包。EAP 数据包的格式如下。

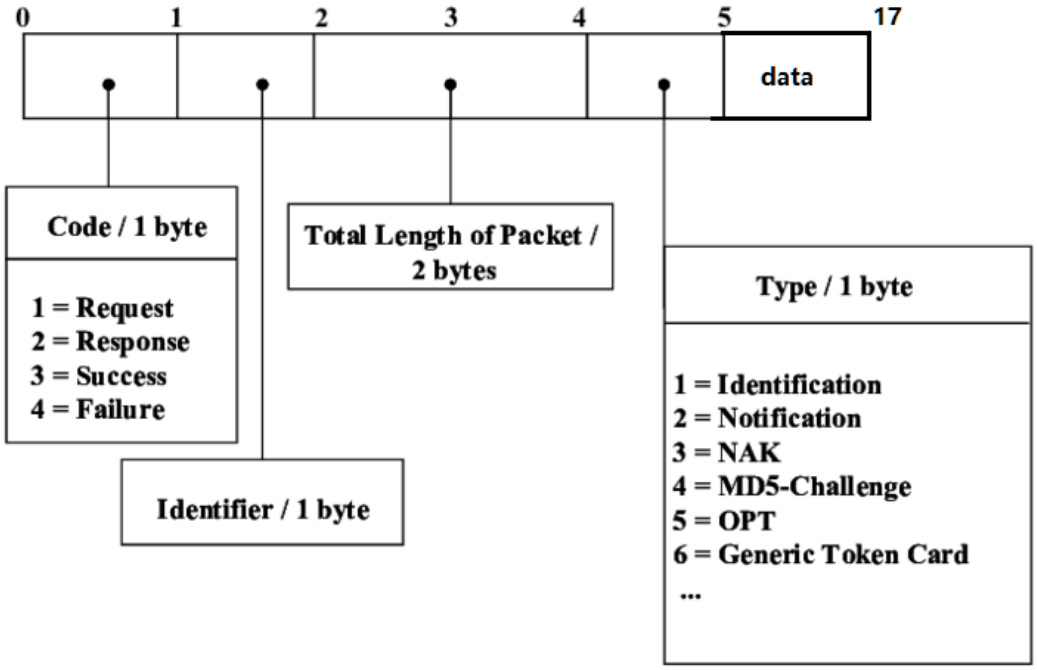


图 3- 9 EAP-Response-Identity 中 EAP 数据包格式

图 3-9 中 Code 的范围对应图 3-7 中的位置在 0x0012，固定值为 02。Code 定义了 EAP 数据包的类型。当 EAP 数据包的类型为 Response 时，值为 02。

图 3-9 中 Identifier 的范围对应图 3-7 中的位置在 0x0013，值为 01。EAP-Response-Identity 数据帧中的 Identifier 需要与 EAP-Request-Identity 数据帧中的 Identifier 对应。

图 3-9 中 Total Length of Packet 的范围对应图 3-7 中的位置在 0x0014-0x0015 之间，固定值为 0011。Total Length of Packet 和图 3-8 中 Packet body length 表示的是同一个长度，即 EAP-Response-Identity 数据帧中封装的 EAP 数据包的长度。

图 3-9 中 Type 的范围对应图 3-7 中的位置在 0x0016，固定值为 01。EAP-Response-Identity 数据包类型为 Identity，因此值为 01。

图 3-9 中 data 的范围对应图 3-7 中的位置在 0x0017-0x0022 之间，值为 020100110132303134323131323120353 9。data 是锐捷认证时使用的用户名，明文传输，根据用户名的不同而改变。在图 3-7 中黑框右侧 wireshark 的翻译中可以看到，用户名为 201421121059。

图 3-7 中 0x0028-0x002b 数据段是申请者的 IP 地址，值为 ca57efad。IP 地址的值由申请者的 IP 地址决定，且被锐捷算法加密过，不是明文传输。具体的算法逻辑见第 4 章 4.1 节。

图 3-7 中 0x002c-0x002f 数据段是申请者的子网掩码（Subnet Mask）地址，值为 0000ffff。子网掩码地址的值由申请者的子网掩码地址决定，且被锐捷算法加密过，不是明文传输。具体的算法逻辑见第 4 章 4.1 节。在多次修改子网掩码地址后分析得知，锐捷认证时不会校验子网掩码地址的正确性。

图 3-7 中 0x0030-0x0033 数据段是申请者的网关（Gateway）地址，值为 ca57ff7f。网关地址的值由申请者的网关地址决定，且被锐捷算法加密过，不是明文传输。具体的算法逻辑见第 4 章 4.1 节。在多次修改网关地址后分析得知，锐捷认证时不会校验网关地址的正确性。

图 3-7 中 0x01a4-0x01af 数据段是认证服务名，值为 c1aacda8bfedb4f8bdd3c8eb。这一段是汉字“联通宽带接入”转换为 GB2312 格式后的值。不同于 EAPOL-Start 数据帧，锐捷认证时校验 EAP-Response-Identity 数据帧中服务名的正确性。

图 3-7 中 0x0241-0x024b 数据段是申请者的域名系统（Domain Name System）地址，值为 3137322e31372e382e3332。从图 3-7 黑框右侧 wireshark 的翻译可知，此 DNS 地址为 172.17.8.32。域名系统的值由申请者的域名地址决定，明文传输。在多次修改域名地址后分析得知，锐捷认证时不会校验域名地址的正确性。

图 3-7 中其余未提到的数据段，目前暂未弄清楚它们的含义和作用。jmuSupplicant 构造 EAP-Response-Identity 数据帧时，保持这些数据段与锐捷客户端一致，不影响认证结果。

以上为对锐捷认证过程中 EAP-Request-Identity 数据帧的分析。

3.4 EAP-Request-MD5-Challenge 数据帧

验证者向申请者发送 EAP-Request-MD5-Challenge 数据帧是锐捷认证的第 4 步。使用 wireshark 软件捕获锐捷认证过程中 EAP-Request-MD5-Challenge 数据帧元数据如下图所示。

```
0000  f0 76 1c 5b cc 7b 00 1a  a9 17 ff ff 88 8e 01 00   .v.[.{.. ........
0010  00 54 01 02 00 54 04 10  59 7e b7 52 c8 4e 65 15   .T...T.. Y~.R.Ne.
0020  7c df 81 16 f9 c5 58 4b  00 00 13 11 2e 03 01 00   |.....XK ........
0030  00 13 11 66 33 d2 c6 b6  af bf ed b4 f8 bd d3 c8   ...f3... ........
0040  eb 40 bd cc d3 fd cd f8  bd d3 c8 eb 40 c1 aa cd   .@...... ....@...
0050  a8 bf ed b4 f8 bd d3 c8  eb 40 b5 e7 d0 c5 bf ed   ........ .@......
0060  b4 f8 bd d3 c8 eb                                 ......
```

图 3- 10 EAP-Request-MD5-Challenge 数据帧

图 3-10 黑框中的数据为 EAP-Request-MD5-Challenge 数据帧的元数据，黑框左侧标注了每行第一个字节数据的位置，黑框右侧是 wireshark 软件自动对黑框中元数据的翻译，便于使用者更直观地理解。

EAP-Request-MD5-Challenge 数据帧格式如下图所示。

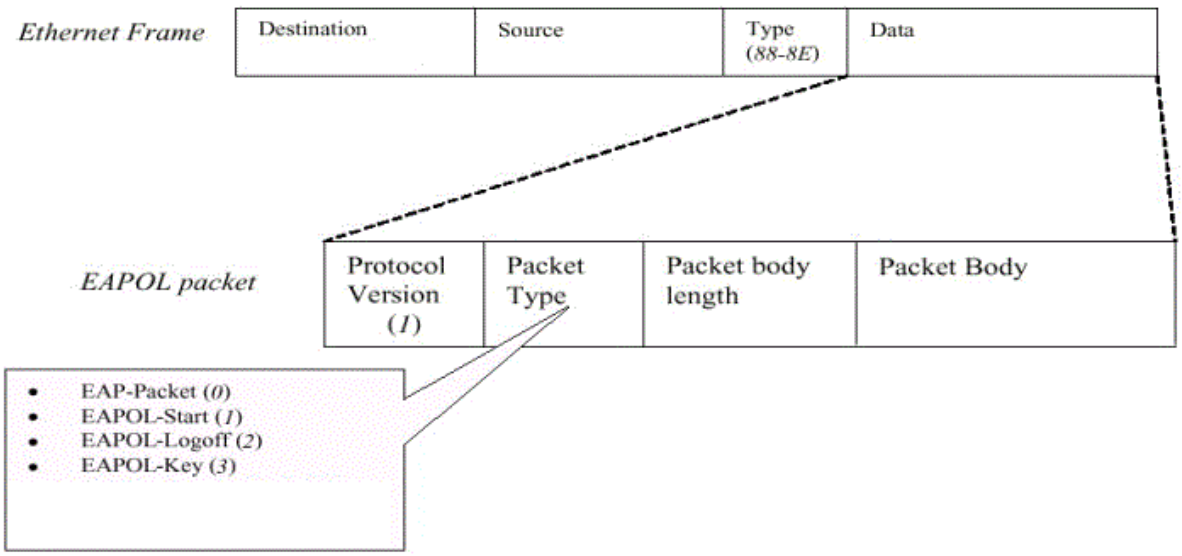


图 3- 11 EAP-Request-MD5-Challenge 数据帧格式

下面将按照图 3-11 中标注的 EAP-Request-MD5-Challenge 数据帧格式，对照图 3-10 中的 EAP-Request-MD5-Challenge 数据帧元数据，按顺序解释锐捷认证时需要用到的各段元数据的含义。

图 3-11 中 Destination 的范围对应图 3-10 中的位置在 0x0000-0x0005 之间，值为 f0761c5bcc7b。Destination 为申请者网络设备的 MAC 地址。

图 3-11 中 Source 的范围对应图 3-10 中的位置在 0x0006-0x000B 之间，值为 001aa917ffff。Source 是验证者网络设备的 MAC 地址，是由验证者的网络设备决定的。

图 3-11 中 Type 的范围对应图 3-10 中的位置在 0x000C-0x000D 之间，固定值为 888e。802.1X 协议的协议号为两个字节的 888e（0x888e）。当以太网帧的 Type 为 888e 时，表示此以太网帧中包含一个 EAPOL 数据帧。

图 3-11 中 Protocol Version 的范围对应图 3-10 中的位置在 0x000E，固定值为 01。Protocol Version 代表 802.1X 认证版本为 802.1X-2001。

图 3-11 中 Packet Type 的范围对应图 3-10 中的位置在 0x000F，固定值为 00。Packet Type 代表当前 EAPOL 的种类为 EAP 数据包。

图 3-11 中 Packet body length 的范围对应图 3-10 中的位置在 0x0010-0x0011 之间，固定值为 0054，Packet body length 表示在 EAP-Request-MD5-Challenge 数据帧中封装的 EAP 数据包的长度。为 0054 意味着封装了 84 个字节的 EAP 数据。

图 3-11 中 0x0012-0x0065 之间的 16 进制数据段为 EAP-Request-MD5-Challenge 数据帧中封装的 EAP 数据包。EAP 数据包的格式如下。

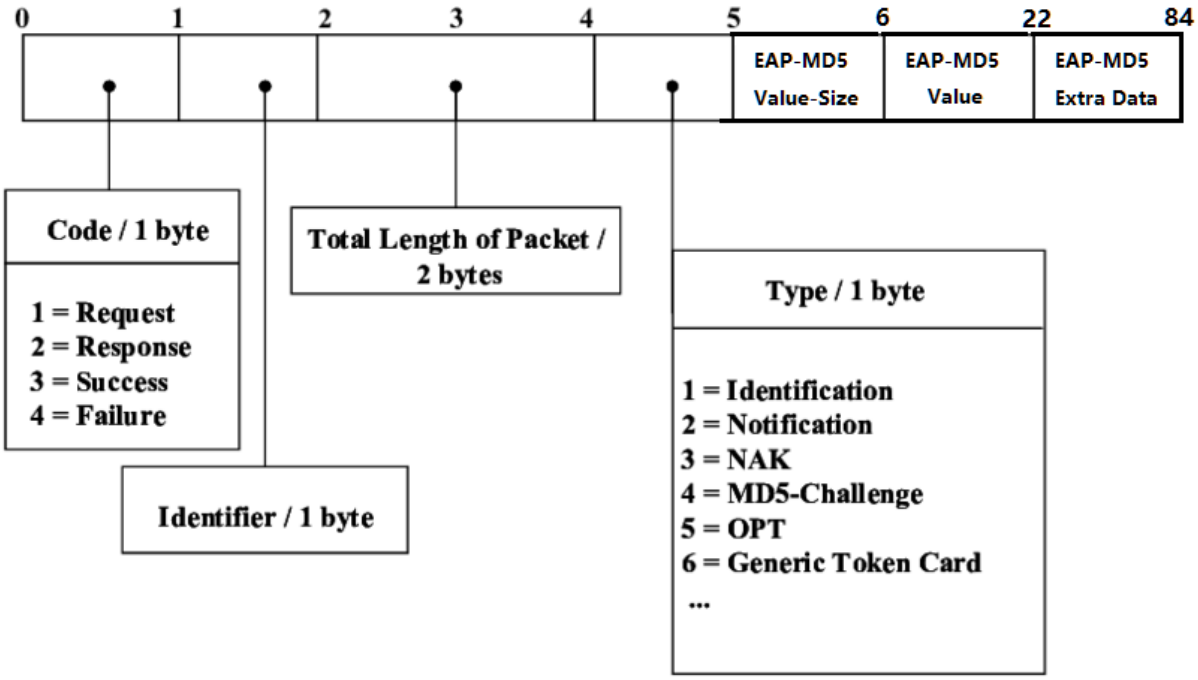


图 3- 12 EAP-Request-MD5-Challenge 中 EAP 数据包格式

图 3-12 中 Code 的范围对应图 3-10 中的位置在 0x0012，固定值为 01。Code 定义了 EAP 数据包的类型。当 EAP 数据包的类型为 Request 时，值为 01。

图 3-12 中 Identifier 的范围对应图 3-10 中的位置在 0x0013，值为 02。若出现 Request 数据帧因超时发生重传时，重传后的 Identifier 要在重传前的 Identifier 值的基础上加 1。此时简化处理，忽略异常情况。

图 3-12 中 Total Length of Packet 的范围对应图 3-10 中的位置在 0x0017，固定值为 10。Total Length of Packet 和图 3-5 中 Packet body length 表示的是同一个长度，即 EAP-Request-Identity 数据帧中封装的 EAP 数据包的长度。

图 3-12 中 Type 的范围对应图 3-10 中的位置在 0x0017，固定值为 04。Type 代表初始 EAP Request/Response 类型（Initial EAP Request/Response Types）。EAP-Request-MD5-Challenge 类型为 MD5-Challenge，因此值为 04。

图 3-12 中 EAP-MD5 Value-Size 的范围对应图 3-11 中的位置在 0x0017，固定值为 10。EAP-MD5 Value-Size 代表 EAP 数据包中存储的 EAP-MD5 Value 长度为 16 个字节。

图 3-12 中 EAP-MD5 Value 的范围对应图 3-10 中的位置在 0x0018-0x0027 之间，值为 597eb752c84e65157cdf8116f9c5584b。EAP-MD5 Value 在每一个 EAP-Request-MD5-Challenge 数据帧中都不同。

图 3-10 中其余未提到的数据段，目前暂未弄清楚它们的含义和作用。

以上为对锐捷认证过程中 EAP-Response-MD5-Challenge 数据帧的分析。

3.5 EAP-Response-MD5-Challenge 数据帧

申请者向验证者发送 EAP-Response-MD5-Challenge 数据帧是锐捷认证的第 5 步。使用 wireshark 软件捕获锐捷认证过程中 EAP-Response-MD5-Challenge 数据帧元数据如下图所示。

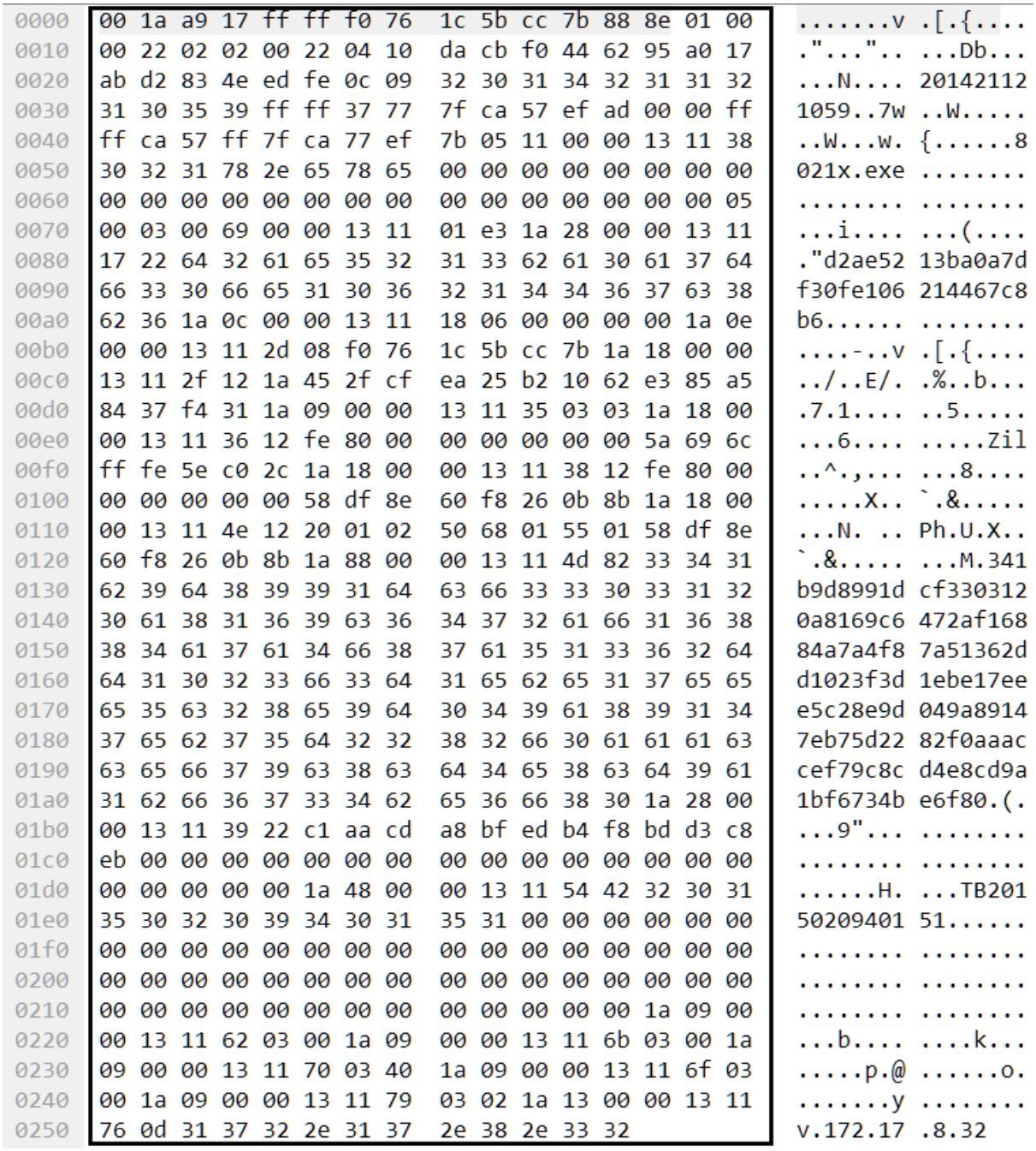


图 3- 13 EAP-Response-MD5-Challenge 数据帧

图 3-13 黑框中的数据为 EAP-Response-MD5-Challenge 数据帧的元数据，黑框左侧标注了每行第一个字节数据的位置，黑框右侧是 wireshark 软件自动对黑框中元数据的翻译，便于使用者更直观地理解。

EAP-Response-MD5-Challenge 数据帧格式如下图所示。

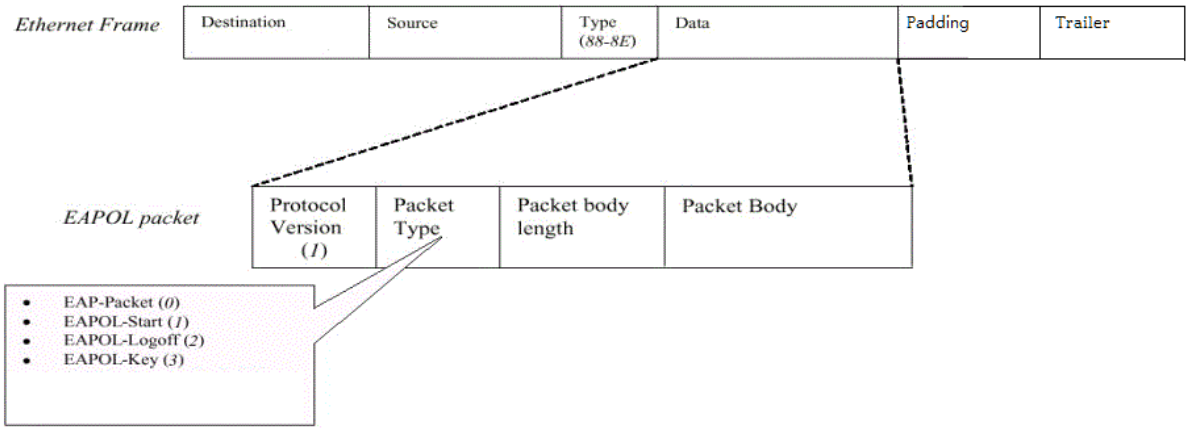


图 3- 14 EAP-Response-MD5-Challenge 数据帧格式

下面将按照图 3-14 中标注的 EAP-Response-MD5-Challenge 数据帧格式，对照图 3-13 中的 EAP-Response-MD5-Challenge 数据帧元数据，按顺序解释锐捷认证时需要用到的各段元数据的含义。

图 3-14 中 Destination 的范围对应图 3-13 中的位置在 0x0000-0x0005 之间，值为 001aa917ffff。Destination 是验证者网络设备的 MAC 地址。

图 3-14 中 Source 的范围对应图 3-13 中的位置在 0x0006-0x000B 之间，值为 f0761c5bcc7b。Source 是申请者网络设备的 MAC 地址。

图 3-14 中 Type 的范围对应图 3-13 中的位置在 0x000C-0x000D 之间，固定值为 888e。802.1X 协议的协议号为两个字节的 888e（0x888e）。当以太网帧的 Type 为 888e 时，表示此以太网帧中包含一个 EAPOL 数据帧。

图 3-14 中 Protocol Version 的范围对应图 3-13 中的位置在 0x000E，固定值为 01。Protocol Version 代表 802.1X 认证版本为 802.1X-2001。

图 3-14 中 Packet Type 的范围对应图 3-13 中的位置在 0x000F，固定值为 00。Packet Type 代表当前 EAPOL 的种类为 EAP 数据包。

图 3-14 中 Packet body length 的范围对应图 3-13 中的位置在 0x0010-0x0011 之间，值为 0022，Packet body length 表示在 EAP-Response-MD5-Challenge 数据帧中封装了长度为 34 个字节的 EAP 数据包。

图 3-8 中 0x0012-0x0033 之间的 16 进制数据段为 EAP-Response-MD5-Challenge 数据帧中封装的 EAP 数据包。EAP 数据包的格式如下。

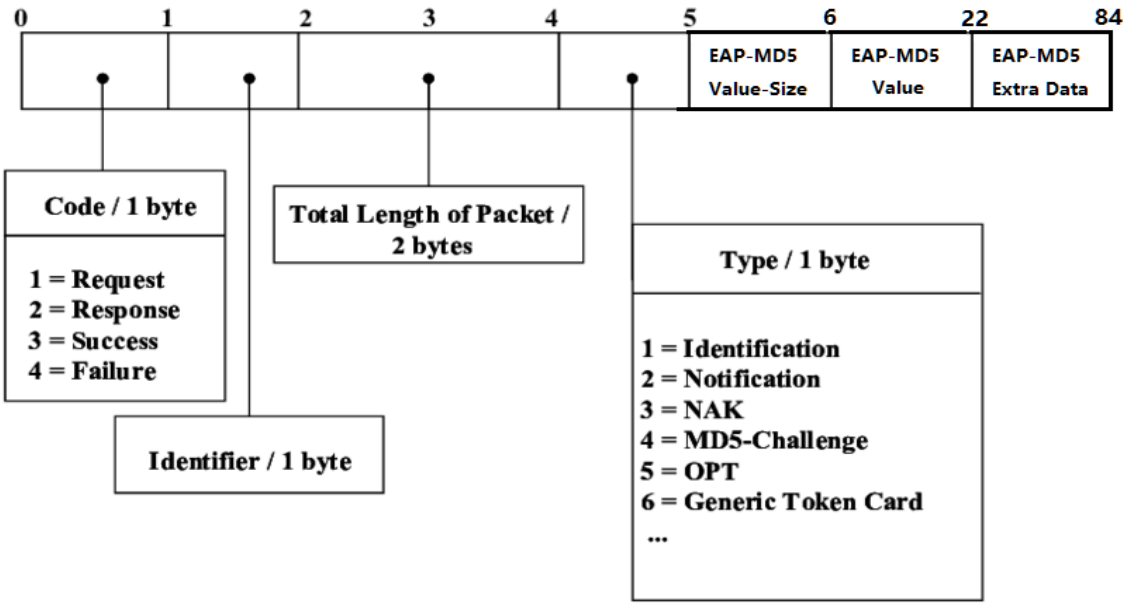


图 3- 15 EAP-Response-MD5-Challenge 中 EAP 数据包格式

图 3-15 中 Code 的范围对应图 3-13 中的位置在 0x0012，固定值为 02。Code 定义了 EAP 数据包的类型。当 EAP 数据包的类型为 Response 时，值为 02。

图 3-15 中 Identifier 的范围对应图 3-13 中的位置在 0x0013，值为 02。EAP-Response-MD5-Challenge 数据帧中的 Identifier 需要与 EAP-Request-MD5-Challenge 数据帧中的 Identifier 对应。

图 3-15 中 Total Length of Packet 的范围对应图 3-13 中的位置在 0x0014-0x0015 之间，固定值为 0022。Total Length of Packet 和图 3-14 中 Packet body length 表示的是同一个长度，即 EAP-Response-MD5-Challenge 数据帧中封装的 EAP 数据包的长度。

图 3-15 中 Type 的范围对应图 3-7 中的位置在 0x0016，固定值为 04。EAP-Response-MD5-Challenge 数据包类型为 MD5-Challenge，因此值为 04。

图 3-15 中 EAP-MD5 Value-Size 的范围对应图 3-13 中的位置在 0x0017，固定值为 10。EAP-MD5 Value-Size 代表 EAP 数据包中存储的 EAP-MD5 Value 长度为 16 个字节。

图 3-15 中 EAP-MD5 Value 的范围对应图 3-13 中的位置在 0x0018-0x0027 之间，值为 dacbf0446295a017abd2834eedfe0c09。EAP-MD5 Value 在每一个 EAP-Request-MD5-Challenge 数据帧中都不同，由固定的算法计算得出。具体的算法逻辑见第 4 章。

图 3-15 中 EAP-MD5 Extra Data 的范围对应图 3-13 中的位置在 0x0028-0x0033 之间，值为 323031343231313231303539。EAP-MD5 Extra Data 是锐捷认证时使用的用户名，明文传输，根据用户名的不同而改变。在图 3-13 中黑框右侧 wireshark 的翻译中可以看到，用户名为 201421121059。

图 3-13 中 0x0039-0x003c 数据段是申请者的 IP 地址，值为 ca57efad。IP 地址的值由申请者的 IP 地址决定，且被锐捷算法加密过，不是明文传输。具体的算法逻辑见第 4 章 4.1 节。

图 3-13 中 0x003d-0x0040 数据段是申请者的子网掩码（Subnet Mask）地址，值为 0000ffff。子网掩码地址的值由申请者的子网掩码地址决定，且被锐捷算法加密过，不是明文传输。具体的算法逻辑见第 4 章 4.1 节。在多次修改子网掩码地址后分析得知，锐捷认证时不会校验子网掩码地址的正确性。

图 3-13 中 0x0041-0x0044 数据段是申请者的网关（Gateway）地址，值为 ca57ff7f。网关地址的值由申请者的网关地址决定，且被锐捷算法加密过，不是明文传输。具体的算法逻辑见第 4 章 4.1 节。在多次修改网关地址后分析得知，锐捷认证时不会校验网关地址的正确性。

图 3-13 中 0x00c4-0x00d3 数据段是一段加密后的数据，本文将其命名为短加密。短加密的值由三段数据作为输入参数运用算法计算后得出。具体的算法逻辑见第 4 章。

图 3-13 中 0x00c4-0x00d3 数据段是一段加密后的数据，本文将其命名为短加密。短加密的值为三段数据运用算法计算后得出。具体的算法逻辑见第 4 章。

图 3-13 中 0x012d-0x01ac 数据段是一段加密后的数据，本文将其命名为 V4 加密。V4 加密的值是锐捷认证数据帧中最复杂的一段加密数据。具体的算法逻辑见第 4 章。

图 3-13 中 0x0241-0x024b 数据段是申请者的域名系统（Domain Name System）地址，值为 3137322e31372e382e3332。从图 3-7 黑框右侧 wireshark 的翻译可知，此 DNS 地址为 172.17.8.32。域名系统的值由申请者的域名地址决定，明文传输。在多次修改域名地址后分析得知，锐捷认证时不会校验域名地址的正确性。

图 3-13 中 0x01b5-0x01c1 数据段是认证服务名，值为 c1aacda8bfedb4f8bdd3c8eb。这一段是汉字“联通宽带接入”转换为 GB2312 格式后的值。不同于 EAPOL-Start 数据帧，锐捷认证时校验 EAP-Response-MD5-Challenge 数据帧中服务名的正确性。

图 3-13 中其余未提到的数据段，目前暂未弄清楚它们的含义和作用。jmuSupplicant 构造 EAP-Response-MD5-Challenge 数据帧时，保持这些数据段与锐捷客户端一致，不影响认证结果。

以上为对锐捷认证过程中 EAP-Response-MD5-Challenge 数据帧的分析。

第 4 章 锐捷认证加密数据算法

4.1 计算 IP 地址、子网掩码、网关

锐捷客户端向验证者传输用户设备 IP 地址、子网掩码、网关时，将数据加密后再进行传输。具体加密算法如下。

输入参数

参数 1：IP 地址，子网掩码或网关

输出参数

加密后的 IP 地址，子网掩码或网关

算法逻辑

参数 1 长度为 4 个字节。将参数 1 每一个字节的 8 比特颠倒并取反后，输出结果

```
Begin 算法开始
  输入 A
  FOR 1 to 4
  颠倒 A 当前字节的 8 比特后取反
  输出 A
End 算法结束
```

4.2 计算 EAP-MD5 Value[9]

此算法在【RFC 1994， Page 8】中有说明，应用于 EAP 协议中。

输入参数

参数 1：16 进制数字数组。由 Identifier（EAP-Request-MD5-Challenge），锐捷认证密码，EAP-MD5 Value（EAP-Request-MD5-Challenge）三段 16 进制数据依次拼接而成。

输出参数

16 进制数字数组，长度为 16 个字节。

算法逻辑

将参数 1 传入 MD5 哈希算法中，提取结果的前 16 位进行输出。

Begin 算法开始

输入 Identifier，锐捷认证密码，EAP-MD5-Value

A = Identifier + 锐捷认证密码+EAP-MD5-Value

B = MD5_Algorithm(A);

输出 B 的前 16 比特

End 算法结束

4.3 计算短加密值

短加密值是一段长度为 16 字节的数据，位于 EAP-Response-MD5-Challenge 数据帧中。在集美大学锐捷认证过程中，认证服务器会校验此段数据的正确性。

输入参数

参数 1：16 进制数字数组。由 EAP-MD5 Extra Data（锐捷认证用户名），EAP-MD5 Value（EAP-Request-MD5-Challenge）两段 16 进制数据依次拼接而成。

参数 2：锐捷认证密码。

输出参数

16 进制数字数组，长度为 16 个字节。

算法逻辑

1、将参数 1 传入 MD5 哈希算法中，提取结果的前 16 位。

2、将步骤 1 的计算结果与参数 2 进行异或运算，将长度为 16 个字节的异或结果输出。

```
Begin 算法开始
  输入 EAP-MD5 Extra Data，EAP-MD5 Value，锐捷认证密码
  A =  EAP-MD5 Extra Data + EAP-MD5 Value
  B = MD5_Algorithm( A );
  C = B 的前 16 比特
  D = C XOR 锐捷认证密码
  输出 D 的前 16 比特
End 算法结束
```

4.4 计算 V4 加密值

V4 加密值是一段长度为 128 字节的加密数据，位于 EAP-Response-MD5-Challenge 数据帧中。锐捷认证时会校验这段数据的正确性。该算法取自”hyrathb/mentohust”开源项目，位于 checkV4.c 文件中。

输入参数

参数 1： EAP-MD5 Value（EAP-Request-MD5-Challenge）

输出参数

长度为 128 字节的 V4 加密值。

算法逻辑

该算法首先取 EAP-MD5 Value（EAP-Request-MD5-Challenge）中两个字节进行计算，然后根据计算结果将 V4 加密值的计算分为 5 种情况。5 种情况均为哈希计算，每种哈希计算的哈希算法不同。

在 5 种情况中的某一种情况的哈希计算完成后，再做一次 Whirlpool 哈希计算，得出最终 128 字节的 V4 加密值。

```
Begin 算法开始
  输入 EAP-MD5 Value
  A = （ EAP-MD5 Value[0]+ EAP-MD5 Value[3] ） % 5u
  Switch(A)
  {
  Case 0:
  rhash_md5_algorithm();
  rhash_md5_algorithm();
  Case 1:
  rhash_sha1_algorithm();
  rhash_sha1_algorithm();
  Case 2:
    rhash_tiger_altorithm();
    ampheck_ripemd128_algorithm();
  Case 3:
```

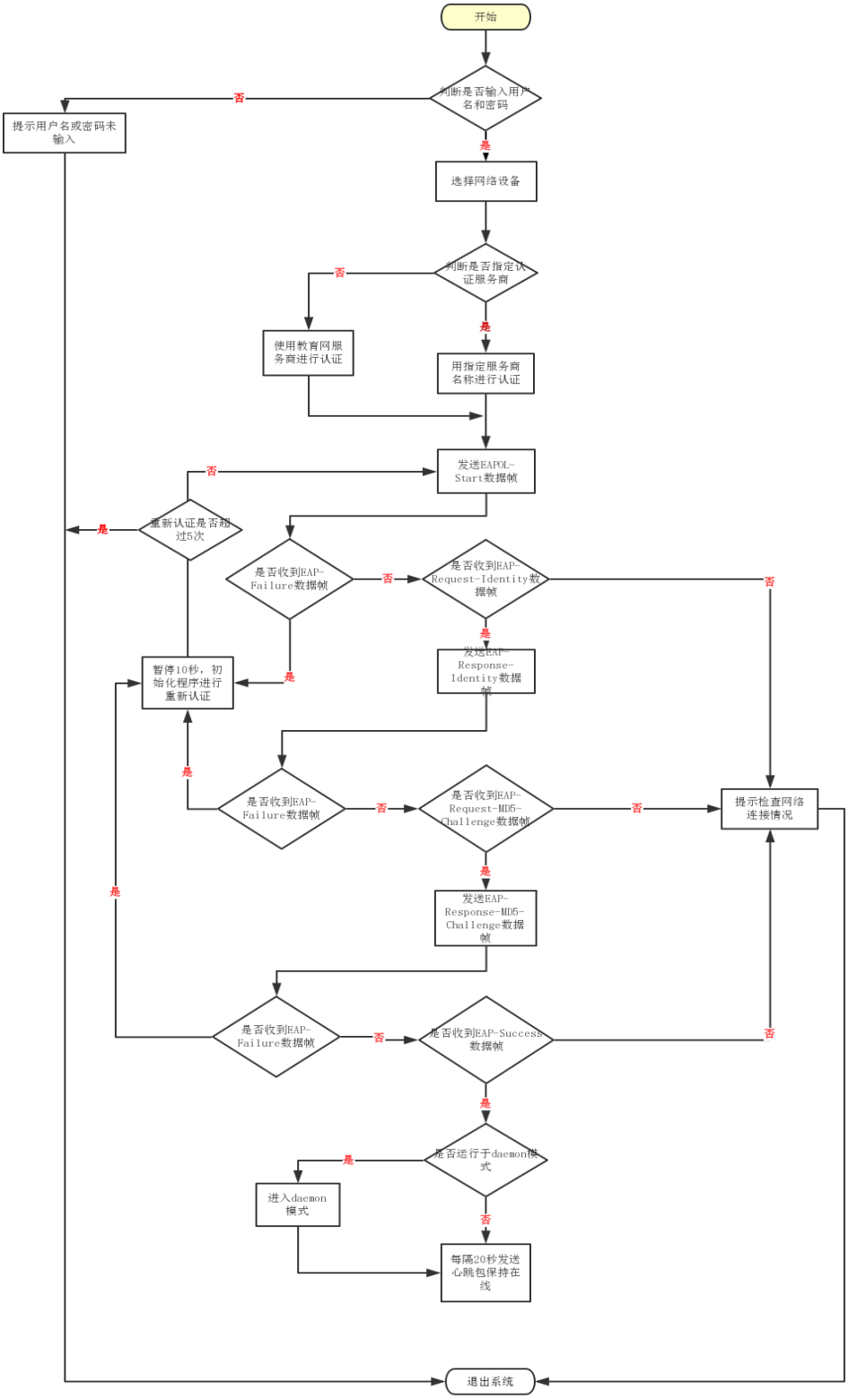
```
    rhash_tiger_algorithm();
    ampheck_ripemd128_algorithm();
  Case 4:
    rhash_tiger_algorithm();
  rhash_sha1_algorithm();
  }
  rhash_whirlpool_algorithm();
```

输出 128 字节的 V4 加密值。

End 算法结束

第 5 章 jmuSupplicant 实现过程

5.1 流程图



5.2 核心功能实现方法

5.2.1 只允许一个 jmuSupplicant daemon 进程运行

默认情况下，jmuSupplicant 仅支持单个进程运行，因此 jmuSupplicant 在运行开始就判断是否已经有 jmuSupplicant 进程正在运行。如果没有，则 jmuSupplicant 继续执行后续认证步骤。如果已有运行中的进程，jmuSupplicant 立刻终止运行。此功能依靠锁文件（File Locking）[10]实现，方法如下。

1、创建锁文件

使用 int open(const char *pathname, int flags);函数，打开或创建指定了路径和名称的锁文件。

```
int fd = -1;
fd = open("/var/run/test_lock.pid", O_CREAT | O_RDWR);
```

2、构造结构体 struct flock，该结构体中的参数描述了锁的行为

```
struct flock lock;
lock.l_type = F_WRLCK;  // F_RDLCK, F_WRLCK, F_UNLCK
lock.l_start = 0;  // byte offset, relative to l_whence
lock.l_whence = SEEK_SET;  // SEEK_SET, SEEK_CUR, SEEK_END
lock.l_len = 0; // #bytes (0 means to EOF)
```

3、使用 int fcntl(int fd, int cmd, ... /* arg */);函数，设置锁。

```
fcntl(fd, F_SETLKW, &lock);
```

经过以上三个步骤，jmuSupplicant 的一个进程创建或打开了锁文件"/var/run/test_lock.pid"，并给这个锁文件加了锁。这个进程将占有这个锁文件直到该进程终止。

当一个新的 jmuSupplicant 进程运行时，该进程会首先判断是否有另外的 jmuSupplicant 进程在占有锁文件"/var/run/test_lock.pid"。方法如下。

1、构造结构体 struct flock，该结构体中的参数描述了锁的行为

```
struct flock lock;
lock.l_type = F_WRLCK;  // F_RDLCK, F_WRLCK, F_UNLCK
lock.l_start = 0;  // byte offset, relative to l_whence
lock.l_whence = SEEK_SET;  // SEEK_SET, SEEK_CUR, SEEK_END
lock.l_len = 0; // #bytes (0 means to EOF)
```

2、使用 int fcntl(int fd, int cmd, ... /* arg */);函数，查看指定的锁文件锁的情况。

```
fcntl(fd, F_GETLK, &lock);
```

3、查看 int fcntl(int fd, int cmd, ... /* arg */)；函数执行后， 结构体 struct flock 中 lock.l_type 变量值。

如果 lock.l_type == F_UNLCK， 代表此时没有另外的 jmuSupplicant 进程占用锁文件，该 jmuSupplicant 进程继续执行后续认证步骤。

如果 lock.l_type != F_UNLCK，代表此时有另外的 jmuSupplicant 进程占用锁文件，该 jmuSupplicant 进程终止。

5.2.2 发送数据帧

此功能依靠 libpcap 库[11]函数实现对底层数据帧的传递。方法如下。

1、使用 pcap_t *pcap_open_live();函数打开网卡，获取一个捕获封包的句柄（handle）。

```
pcap_t* descr = pcap_open_live(const char *device, int snaplen, int promisc, int to_ms, char *errbuf）;
```

2、使用 pcap_inject();函数通过网卡发送数据帧。

```
pcap_inject(pcap_t *p, const void *buf, size_t size);
```

5.2.3 接收数据帧

此功能依靠 libpcap 库函数实现对底层数据帧的接收。方法如下。

1、使用 pcap_t *pcap_open_live();函数打开网卡，获取一个捕获封包的句柄（handle）。

pcap_t* descr = pcap_open_live(const char *device, int snaplen, int promisc, int to_ms, char *errbuf）;

2、使用 pcap_compile();函数编译字符串到过滤程序中。该字符串用于指定从网卡捕获 802.1X 协议的数据帧。

pcap_compile(pcap_t *p, struct bpf_program *fp,const char *str, int optimize, bpf_u_int32 netmask);

3、使用 pcap_setfilter();函数给捕获封包的句柄指定过滤程序；

pcap_setfilter(pcap_t *p, struct bpf_program *fp);

4、使用 pcap_loop()[12];函数从网卡捕获数据帧，传入回调函数中处理。

int pcap_loop(pcap_t *p, int cnt, pcap_handler callback, u_char *user);

5.2.4 夜晚断网期间登陆

校园夜间断网的限制，不包括部分校园办公区域。申请者向验证者请求认证时，将自己的 IP 地址放入认证数据帧中，以表示申请者认证时所在的认证区域。因此，jmuSupplicant 通过伪造办公区域 IP 地址进行认证（服务名只能为‘教育网接入’），即可实现夜间断网后认证成功。

此时，由于伪造的是办公区域 IP 地址，验证者为申请者打开的端口对应的是伪造的办公区域 IP 地址，申请者实际的 IP 地址对应的验证者端口依然是被限制的。为了让实际的 IP 地址解除限制，jmuSupplicant 模拟锐捷客户端进行切换服务操作，虽然服务无法切换成功，但是在这之后，验证者会将申请者实际 IP 地址对应的端口打开，自此申请者真正能够访问互联网和其他区域网络中的资源。

锐捷客户端切换服务操作，实际是让申请者重新进行认证，从发送 EAPOL-Start 数据帧给验证者，到最终收到 EAP-Success 数据帧的一个完整的认证过程。值得注意的是，切换服务操作时申请者发送的 EAPOL-Start 数据帧和 EAP-Response-MD5-Challenge 数据帧的尾部数据比普通认证时的更长，并保持固定。猜测锐捷是根据尾部数据来判断申请者认证方式为普通认证还是切换服务时进行的的重新认证。

Begin 算法开始

jmuSupplicant 将办公区域 IP 地址填入认证数据帧中，使用教育网接入服务，进行认证

认证成功后。JmuSupplicant 将实际 IP 地址填入认证数据帧中，使用联通快带接入服务，再次认证

收到验证者发送的 EAP-Success 数据帧，夜晚断网期间登陆成功

End

5.3 交叉编译[13]到路由器中使用

以下步骤中介绍的目标路由器芯片型号为 mt7620，路由器操作系统版本为 PandoraBox R2 14.09。

1、下载 OpenWrt 软件开发工具包（SDK）

该工具包中有对应 mt7620 芯片的编译器 mipsel-openwrt-linux-gcc，我们需要用这个编译器将 jmuSupplicant 编译成路由器中能够运行的程序。

该工具包中有对应 mt7620 芯片的 libpcap.a 静态链接库，jmuSupplicant 程序部分功能依靠 libpcap 库函数才能完成，因此需要将 libpcap.a 进行链接后才能够生成运行于路由器的程序。

该工具包中有对应 mt7620 芯片的 libiconv.a 静态链接库，jmuSupplicant 程序部分功能依靠 libpcap 库函数才能完成，因此需要将 libiconv.a 进行链接后才能够生成运行于路由器的程序。

2、修改 Makefile 文件，指定编译器为 mipsel-openwrt-linux-gcc。

3、修改 Makefile 文件，将 libpcap.a 和 libiconv.a 添加到链接文件中。

4、修改 Makefile 文件，添加头文件 pcap.h， iconv.h 到头文件依赖中。

5、执行 make -f Makefile 命令，路由器中可运行的 jmuSupplicant。

第 6 章 总结与展望

6.1 总结

本文分析了锐捷认证过程中申请者与验证者之间往来的数据帧内容及格式，解释了锐捷加密数据段的算法，并简要说明了 jmuSupplicant 核心功能的实现方法和将 jmuSupplicant 交叉编译到芯片型号为 mt7620 路由器上的步骤。

了解认证数据帧的内容及格式，能够从核心的层次理解锐捷认证过程中的行为。本文选取锐捷认证过程中核心的 5 个数据帧 EAPOL-Start，EAP-Request-Identity，EAP-Response-Identity，EAP-Request-MD5-Challenge，EAP-Response-MD5-Challenge 进行详细介绍，目的不仅是介绍锐捷认证时的行为，而且要说清楚锐捷认证发送和接收的数据帧中都有什么数据， 这些数据有什么含义。这些信息是通过 wireshark 软件捕获大量的锐捷认证数据帧，比对它们之间的不同点，分析特定数据段的变化，再加上分析锐捷认证日志得到的。

了解锐捷加密数据段的算法，是成功实现 jmuSupplicant 的一个关键所在。清楚算法的细节，才能将认证信息正确地加密后传输给验证者，完成认证过程。

6.2 展望

成功实现适配集美大学第三方锐捷认证客户端，我感到很高兴。但仍有一些遗憾。

1、没能弄清楚 V4 客户端校验算法的实现逻辑，希望能和 hyrathb/mentohust 项目作者请教一下他是如何分析出 V4 客户端校验算法的。

2、没能实现 jmuSupplicant 在路由器上长时间的网络汇聚。路由器支持多 WAN 口网络汇聚功能，当路由器启用两个 WAN 口后，我能够用 jmuSupplicant 分别在两个 WAN 口上认证成功。可是每次第二个认证的锐捷账号无法保持在线，只能持续 4 分 10 秒。如果这个功能能够实现的话，网络带宽理论上能够提升 2 倍，十分遗憾。

致谢

临别之际，心中充斥着留恋与不舍。漫步在集大的各个角落，眼中的每一处花、草、建筑，似乎都在挥手宣告属于我的本科时代即将终结。情有多种，离情最苦。

感谢黄斌老师在我日常学习和生活中给予的帮助，以及在我撰写这篇论文的过程中给予的关心与监督。从黄斌老师的工作中，我能感受到他认真的态度以及对学生真诚的关爱之情。黄斌老师的师风让我由衷的敬佩。

感谢班主任汪志华老师 4 年来给予我的关心和照顾。汪志华老师在每次班会或聚会时捎带给同学们的巧克力，让我觉得那是世界上最好吃的巧克力。

感谢林颖贤老师对我的支持与鼓励。大学里上过最多的课就是林颖贤老师的课，林颖贤老师每次见到我就像老朋友一样，询问近况，对我的想法提出宝贵的建议。我很开心能在大学里遇到亲切温柔的林颖贤老师。

感谢李旺老师在专业技术上对我的引领。体会过李旺老师的授课模式后，大幅提升了我对计算机科学的理解层次。

感谢曾勇进老师在我思维模式提升上的帮助。曾勇进老师帮助我度过了一段迷茫期，纠正了我在行为处事上的一些缺点，开启了我在理财上的尝试。

感謝臺灣中原大學資訊工程學院的吳宜鴻、張元翔、楊明豪、鄭憲永等老師的支持和幫助。那一個學期的學業艱難又夢幻，難忘且美好。

感谢关心我的朋友和同学们。是优秀的你们督促我不断进步，因为你们的存在我变得更好。

最后要特别感谢我的爸爸妈妈，感谢你们毫无保留地支持我。

参考文献

[1] 中国有哪些大学在用锐捷认证？". *Zhihu.Com*, 2013, https://www.zhihu.com/question/21517904. Accessed 3 June 2018.

[2] Brown, Edwin Lyle. *802.1 X port-based authentication*. CRC Press, 2006.

[3] Lamping, Ulf, and Ed Warnicke. "Wireshark user's guide." *Interface* 4.6 (2004).

[4] Aboba, Bernard, et al. "RFC 3748-Extensible authentication protocol (EAP)." *Network Working Group* (2004).

[5] 锐捷、赛尔认证 mentohust". *Wiki.Ubuntu.Org.Cn*, 2018, http://wiki.ubuntu.org.cn/%E9%94%90%E6%8D%B7%E3%80%81%E8%B5%9B%E5%B0%94%E8%AE%A4%E8%AF%81MentoHUST. Accessed 3 June 2018.

[6] Hyrathb/Mentohust". *Github*, 2015, https://github.com/hyrathb/mentohust. Accessed 3 June 2018.

[7] 深入了解校园网 802.1X 认证的 eap 协议(1)——EAP 的总体流程 - 无证程序员的 PT 桑". *Blog.Ptsang.Net*,

[8] GB2312 (Simplified Chinese) Character Code Table". *Ash.Jp*, 2001, http://ash.jp/code/cn/gb2312tbl.htm. Accessed 2 June 2018.

[9] Simpson, William Allen. "PPP challenge handshake authentication protocol (CHAP)." (1996).

[10]檔案鎖定（File Locking） - CSDN 博客". *Blog.Csdn.Net*, 2006, https://blog.csdn.net/zqy2000zqy/article/details/1137905. Accessed 2 June 2018.

[11]Pcap(3): Packet Capture Library - Linux Man Page". *Linux.Die.Net*, https://linux.die.net/man/3/pcap. Accessed 2 June 2018.

[12]Manpage Of PCAP_LOOP". *Tcpdump.Org*, 2017, https://www.tcpdump.org/manpages/pcap_loop.3pcap.html. Accessed 2 June 2018.

[13]Openwrt Project: Cross Compile". *Openwrt.Org*, 2018, https://openwrt.org/docs/guide-developer/crosscompile. Accessed 2 June 2018.