清 华 大 学

综合论文训练

题目：一种无线物联网的设计和实现

系　　别：电子工程系

专　　业：电子信息科学与技术

姓　　名：王亭午

指导教师：李星 教授

2016 年 6 月 16 日

关于学位论文使用授权的说明

本人完全了解清华大学有关保留、使用学位论文的规定，即：学校有权保留学位论文的复印件，允许该论文被查阅和借阅；学校可以公布该论文的全部或部分内容，可以采用影印、缩印或其他复制手段保存该论文。

(涉密的学位论文在解密后应遵守此规定)

签　名：王靳午　导师签名：李星　日　期：2016年5月30日.

中文摘要

无线物联网旨在构建一个通过无线进行上下游连接，信息交换和控制的物联网络。无线物联网不同于现有的普通网络。在物联网中提供服务和接受服务的对象不再局限于功能齐全的主机。任何嵌入了无线连接和信息交换功能的设备，都可以成为物联网的组成成分。物联网中物体庞大的连通性和信息沟通能力，使得使用者可以在楼宇自动化，智能表，医院智能化等运用上做出许多进步。然而，伴随着无线物联网的巨大运用，巨大的设计挑战也随之而来。无线物联网中的设备可能是非常繁多的，因此在地址分配上，传统的 IPv4 地址分配，在分配能力和双向联通上，都有巨大的缺陷。繁多的设备也对系统的自动配置提出了新的要求。同时，无线物联网中设备有限的计算能力和物联网众多设备工作的复杂无线环境，使得无线传输时的安全性和稳定性都更加难以控制。本系统根据现有无线物联网中遇到的几个主要问题，设计了一套基于 IPv6 的无线物联网。通过使用隧道技术和 IVI 地址分配，在解决双向访问的同时，系统设计考虑了 46 地址过渡的问题。在这个系统中，通过对设备的自动配置机制的设计，无线设备可以自动接入网络并且配置自己的地址信息。

关键词：无线；物联网；IPv6；自动配置；

ABSTRACT

Wireless Internet of Things is aiming to build a network where uplink and downlink connections, information changes and controls are done via wireless network. Wireless Internet of Things is unlike traditional networks that are currently available. In a Wireless Internet of Things, objects that provide or request for a service are not limited to computer hosts that have complete functions. Any embedded devices with wireless connection and the ability of information exchange, could be part of the Internet of Things. The great connectiveness and the ability of exchanging information of the Internet of Things have brought countless progress in the areas such as smart building, smart meters and smart hospital. However, in companion with the many implementations of the Internet of Things are the many challenges of design. Devices could be numerous in a Wireless Internet of Things, because of which, traditional algorithms of allocating IPv4 addresses might be very flawed in the aspect of allocating ability and diconnection. The many devices have also made new demands on the systems' auto-configuration ability. At the same time, the limited calculating ability in a device of Wireless Internet of Things and the complex wireless environment caused by the many devices, have further made security and stability in a wireless transmission more uncontrollable. We design a Wireless Internet of Things structure, based on the major problems met in current Wireless Internet of Things. By using tunnel techniques and IVI address allocation, we consider the IPv4-IPv6 transition while providing diconnection by using IPv6 addresses. In the systems, by designing auto-configuration of the devices, a wireless device can auto-connect the network and auto-configurate its address information.

Keywords： wireless; Internet of Things; IPv6; auto-configuration

目录

第 1 章　引言 1
1.1 选题背景介绍 1
1.2 毕业设计目标和论文的主要结构 2
第 2 章　现有无线物联网设计和比较 4
2.1 现有无线物联网设计和比较 4
2.2 相关工作 5
第 3 章　无线物联网设计场景和框架 8
3.1 运用设计场景 8
3.2 设计框架 8
3.2.1 底层平台设计 8
3.2.2 网络二三层设计 9
3.2.3 物联网网络结构设计 10
3.3 6over4 隧道的基本原理 12
3.4 IVI 地址的基本原理 14
第 4 章　无线物联网设计细节 16
4.1 AP 物联网设备配置 16
4.1.1 建立隧道连接 16
4.1.2 获取与管理地址空间 16
4.1.3 开启 WLAN 服务 18
4.2 Client 物联网设备配置 19
4.2.1 Client 工作流程设计 19
4.2.2 Web 服务器 20
4.2.3 连接状态汇报 22
4.3 AP/Client 物联网设备配置 23
4.4 NMS 配置 24
第 5 章　IVI 地址分配机制 26
5.1 IVI 地址分配机制 26

5.2.1 地址分配机制 27
5.2.2 地址取回机制 28
第 6 章 部署情况和总结 29
6.1 部署情况 29
6.2 网络性能测试 31
6.2.1 系统连通性和路由 31
6.2.2 系统网络测试 32
6.2.3 补充性数据 33
6.3 总结和创新点 34
6.4 下一步工作 35
插图索引 36
参考文献 37
致谢 40
声明 41
附录 A 外文资料的调研阅读报告或书面翻译 42

第1章　引言

1.1 选题背景介绍

和传统的网络更多的将注意力放在完善的主机和服务器不同，物联网将网络的对象进一步推广到了世界上所有的具有信息功能的物理实体。通过将电表，传感器，灯泡等物理器件智能化连入物联网，网络可以实现智能电表控制，楼宇自动化，医院自动化等一系列功能目标[1]。物联网相比传统网络，带来了更多的信息交互可能，但是同时也带来了许多设计上的困难。

物联网的设备，在数量上远远超过传统网络。物联网需要连接和管理的结点不再仅限于传统意义上的电脑，而是包括数量庞大的传感器和控制器在内的所有潜在智能设备。根据爱立信公司的估计，一个成熟的物联网需要考虑的节点数目将大于 500 亿个[2]。这给传统的使用 IPv4 的网络，带来了非常大的挑战。IPv4 能够提供的公网地址数量连成熟物联网结点数量的十分之一都无法达到。在现阶段的研究中，针对 IPv4 的地址缺乏，最为常用的方法是网络地址转换（NAT）方法[3]。然而这个方法在物联网中存在很多的问题。其中一个问题就是双向连通性。通过 NAT 传输的对话是单向的[4]。因此公网无法直接访问处在内网地址下的物联网设备。在物联网的实现中，迅速直接的建立起控制器到设备的对话，从而对设备进行控制和信息交流可以带来很多的便利。快速简洁的端到端连接设计，是物联网设计的一个目标。

同时，在物联网环境下，原有的地址配置也存在不适应性。现有的地址分配方法可以分为静态配置和动态配置两种[5]。在传统的网络静态配置中，设备通过手动配置地址信息（比如静态地址，网关和子网掩码），从而使得设备获得一个固定的通信标识。在物联网中，静态配置是不切实际的。因为设备的数量往往非常庞大。而动态配置，则存在无法将地址和设备捆绑，地址无法追踪和固定的问题。同一个设备在断电重连或者修复后，可能出现获取地址不一致的问题。这给物联网控制带来很大的不便。同样因为物联网设备的不稳定连接，连接检测也成为了一个需要考虑的因素。对设备连接状态的定期监控，可以有效的帮助维护物联网并且保持物联网功能的正常运行。

除此之外，物联网设备的安全性问题是另一个需要考虑的问题。物联网设备的电源和传输能力一般都是有限的，部署的环境一般更加复杂。因此，物联网设备更加容易受到攻击[6]。在物联网中，比较常见的攻击手段包括节点捕获，洪泛攻击（flooding attack），拒绝服务攻击（DoS attack），欺骗攻击（spoofing attack），垃圾信息（spam messages），中间人攻击等等[7]。

1.2 毕业设计目标和论文的主要结构

可以看到，物联网带来了很多新的可能性，但是在设计中，有许多需要解决的问题。本次毕业设计目标如下：（1）实现一个功能成熟的无线物联网框架。在这个物联网框架中，实现物联网设备的智能连接和配置。所有的设备可以实现与外网可靠，快速的连接。（2）兼容 IPv4-IPv6 过渡技术。考虑到 IPv4 在将来的一段时间内依然会是互联网的主要支持协议，因此物联网在设计构想中，应该支持 IPv4 的过渡技术。通过 46 过渡技术，一方面可以将使用 IPv6 作为主要协议的物联网部署在原生并不支持 IPv6 的网络环境中。另一方面，设计框架给管理者或者访问者提供一个纯 IPv4 的网络环境下访问 IPv6 物联网设备，从而实现对设备的控制的访问接口。（3）实现一个用户友好，可靠性稳定性高的网络管理服务器。一方面，通过网络服务器，管理者可以监控下属设备的运行和连接状态。另一方面通过这个控制服务器，管理者可以控制物联网下联所有设备的状态。在服务器中提供多个方便简洁的访问和控制接口。（4）物联网设备数据管理。本项目设计运用场景包括了智能电表。因此在设备上需要实现智能电表的数据搜集和管理功能。并且在设备上提供一个功能齐全的数据网页服务器。（5）实现物联网的安全通信。物联网设备的安全性是一个非常重要的考量。在没有安全性保证时，通信很可能被轻易的窃取甚至被破坏。恶意的设备可能通过侵入物联网，对网络的正常运行和数据的可靠性造成巨大的破坏。（6）进一步开发的友好性。在毕业设计中所搭建的物联网不仅仅可以实现现有的需求，解决现阶段面对的实际问题，还能在后续的开发中进行功能的拓展。

本文的剩余部分结构如下：第二章本文将介绍现有的无线物联网的设计工作，分析它们的优缺点和设计细节。第三章中，具体运用场景和场景中出现的问题将会被介绍和分析。并且在这一章中本文将提出整体的设计方案，详细的说明各个物联网节点自动配置的设计和连接关系。并且简单的介绍一些系统组成部分的原理。第四章本文将对 IPv4-IPv6 地址过渡方案和设计的 IVI 地址分配机制进行分

析和讲解。在第五章中，本文将展示最终的设计结果，做出分析和评价，并且给出结论和下一步可能的工作。

第2章　现有无线物联网设计和比较

2.1 现有无线物联网设计和比较

现有的物联网设计主要运用场景包括医院中的医护和医疗设备自动化、楼宇自动化、包括智能电表和智能电网在内的工业自动化、环境监控和天气预报等等[8]。

在网络的二层设计上，基于 Ad Hoc 模式 Infrastructure（基础架构模式）模式的物联网设计都有运用[9]。在文献[10]中，作者对物联网 Ad Hoc 的设计进行了详细的讨论。使用 Ad Hoc 模式的物联网更加适合结点位置复杂和信道复杂变化的情形。当设备因为信道恶化无法连接或者设备失效的时候，其余的设备可以通过动态更新路由和连接结构，保持传输的顺利。但是 Ad Hoc 模式带来的负面影响也是非常显著的。首先，正如前文中所提到的，物联网设备的数量是非常庞大的。因此 Ad Hoc 模式的设备间通信干扰比较严重。设备间的复杂互联使得路由存在更多冗余，带来更加多的额外无线开销。同时，Ad Hoc 的一个巨大缺陷在于它可以支持的设备数量是有限的。在[11]中，作者的研究结果显示，随着参与到 Ad Hoc 网络设备的数量上升，Ad Hoc 的性能开始迅速下降。和基础架构模式相比，Ad Hoc 的另外一个问题在于安全性。在[10]中可以看到，Ad Hoc 网络的结构更加动态，无线连接情况更加难以监控。因此 Ad Hoc 网络在安全性上，相比较基础架构网络有着先天的不足。

在三层设计上，越来越多的基于 IPv6 的物联网开始出现[12-13]。为了解决 IPv4 地址空间太小的问题，互联网工程任务组（IETF）在上个世纪开始开发 IPv6 协议来作为下一代的网络层协议[14]。IPv6 的地址由 128 位组成，理论上可以为所有的物理设备都分配可用的全局双向访问地址。除此之外，IPv6 也可以应对物联网中复杂大规模路由问题。在 IPv6 的路由协议中，路由表的大小可以得到缩减[15]。同时，上文中已经提到，在物联网中，地址的配置是非常复杂的。而 IPv6 地址在设计之初，就已经考虑了自动配置的问题，因此在物联网设计中，具有天然的优势。

可见，IPv6 因为其足够的地址空间和自动配置能力，适用于多种情况下的物联网设计。在现有的设计中，常见的有以下几种设计[13]：（1）不与外网连接的

物联网 WLAN。在这种情况中，物联网设备通过本地链路 IPv6 地址（Link-local Address）或者内网 IPv6 地址来进行通信。这种情况下的设备需要能够区分自己和其他结点，常使用静态地址分配。（2）物联网设备接入基于 IPv6 的 WLAN，获取公网 IPv6 地址（Globally Unique Address），通过物联网网管直接接入 IPv6 外网从而实现双向联通。此处的物联网网管负责处理物联网设备和外网的路由。通常情况下，物联网设备使用同一个子网。如果下联的设备使用不同的子网前缀，路由会变得更加复杂。（3）物联网设备接入基于 IPv6 的 WLAN，获得一个本地 IPv6 地址（Unique local Address）。设备的请求通过物联网代理（proxy）传达给外网。除此之外，大多数的 IPv6 物联网的设计都是基于前面几种拓扑结构的变种。不同的物联网设计，往往和不同的运用场景和运用需求相关联。

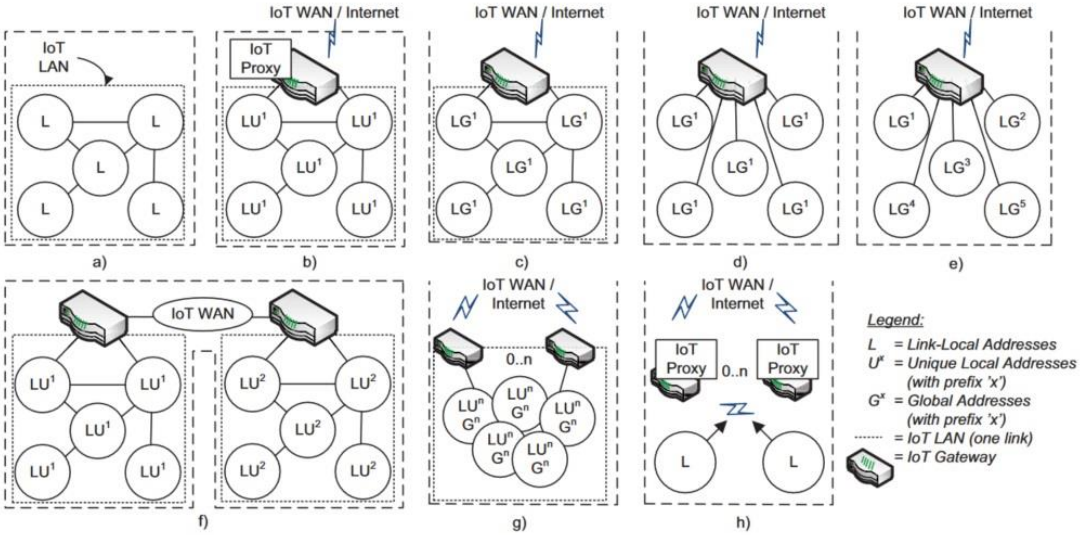

图2.1 几种常见的无线物联网结构图，图片来自[13]

2.2 相关工作

在工业和学术界中，许多的物联网应用已经被开发或者研究。

在[16]中，来自维也纳科技大学的几位学者研究了智能楼宇设计中的无线物联网 IPv4 和 IPv6 运用。在这个工作中，他们对现有的算法进行了综述和总结，同时在搭建物联网系统时尝试利用 IPv6 的新特性。在这个工作中，他们的将楼宇物联网分为三个层次：管理层（management tier），自动层（automation tier）和

接入层（field tier）。在接入层，设备可以通过基于 IP 的方式从（无线）访问接入点（access point）接入物联网。但是使用这个方法必须对每一个物联网设备重新编写 IP 协议栈。同时通信也可以通过基于电力线，无线电广播的非 IP 的方法来实现。作者在考察了现有的基于 IPv4 的主干物联网网络后，提出应该逐渐使用基于 IPv6 的主干网来逐渐进行替换。在文中，他们提出了一个使用 IPv6 的楼宇自动化网络（building automation systems network）的运用。

能耗是无线物联网中的一个常见问题。现有的解决方案基本可以总结成三个方向。在第一个方向中，降低能耗是研究者的主要研究点。第二个方向中，使用绿色能源，搜集外界能量来保持设备的工作是主要的研究点。在最后一个方向中，默认前提是设备更换电池 [17]。在这种情况下，保证在设备耗尽电量关闭之后数据传输的稳定和设备更换电池重新连接后状态的一致是主要的研究重点。

在[18]中，几位来自思科公司和瑞典科学院的作者，充分的探讨了低能耗下 IPv6 物联网的各种设计框架。在文中，几位作者使用了 Contiki 操作系统搭建了一个低能耗的轻量级 IPv6 物联网演示系统。为了实现低能耗的设计目标，几位作者在一个标准的 IPv6 物联网框架下加入了新的节能数据收集协议和无线电责任循环协议（radio duty cycling protocols）。

在[19]中，几位来自瑞士的作者探讨了全球物联网 IPv6 组网的问题和可能的实现。现有的物联网设计往往暂时只考虑了一个系统下设备的配置和通信，而没有考虑全球环境下的巨型物联网设计情形。在全球物联网的背景下，网络尤其是无线网络的异构型会使得搭建一个物理网更加复杂。对于用户来说，访问接口应该遮蔽住网络的异构环境。

在[20]中，轻量级的移动 IPv6 物联网组网是考虑的重点。此文作者主要考虑了医院中物联网的应用，主要探讨了物联网的鲁棒性和能耗。在这个框架中，作者使用 IPSec 来对医院物联网的安全性进行控制。轻量级移动物联网设备的可用内存和能耗都受到极大的限制，因此作者设计了轻量级的移动 IPv6 协议来解决轻量级设备遇到的限制问题。

在[21]中，来自意大利的学者探讨了高度异构化的无线传感网络中，可能的各种物联网架构。在对比了现有的很多私有和非私有的解决方案后，作者将目光投向了最近出现了 6LoWPAN 方案。通过这个方案，可以实现无线传感器通过 IP 协议接入物联网的需求。但是这个方案的问题在于，大量不支持 IP 架构的传感器已经被部署在了生产环境中，因此很难直接进行协议的更新和替换。这篇文章研究的重点是如何兼容老式的互联网架构。在尝试兼容的努力中，作者使用了楼

宇自动化作为测试的运用背景。这篇文章对现有的基于 IP 的物联网方案进行了详细的综述，并配合底层协议进行了分析讲解。本文已经由我进行了中文翻译。具体的翻译可以在这篇毕业论文的最后找到。

第3章　无线物联网设计场景和框架

3.1 运用设计场景

从上文中可以看到，不同的物联网设计方案适用于不同的运用场景。因此，谨慎的分析运用场景需求是非常重要的。

本次毕设的物联网拥有一个实际的运用场景：智能电表。现考虑多个竖井，每个竖井中需要放置若干个智能电表对环境进行监控。在运用场景的时候，需要注意以下几个问题:（1）如何实现对电表直接进行访问，控制和信息交换的功能。（2）如何确定电表在物联网上的网络位置。在竖井中的电表电量可能是很有限的，电表本身和无线网卡也有可能发生损坏。在断电重连或者修复重启后，现有的常见地址分配技术会使得管理者无法确认电表对应的 IP。（3）如何确保电表的数据不被监听窃取。如何确保非法电表无法接入物联网。（4）如何监控连接电表的连接状况。

可以看到，运用需求非常的复杂繁多。设计场景的挑战实际上具有代表价值。在这个毕设中设计的无线物联网框架，对其他不同运用场景下的物联网设计有很大的启发意义。

3.2 设计框架

根据上文提到的 IPv6 优点，在本项目中设计了如图 3.1 的无线物联网网络。在这个图中，每一个 π 都代表一个物联网设备。按照网络设计框架由底层到顶层的顺序，下文首先介绍一下设计框架的一些基本构想。

3.2.1　底层平台设计

在底层设备的选择上，使用了基于 raspbian 操作系统的树莓派。树莓派由慈善组织“Raspberry Pi”发售。树莓派使用 SD 卡来作为自己的硬盘，原则上可以在树莓派上烧些基于 linux 的多个不同操作系统发型版本。比如 OpenWrt，Arch Linux 等等。在设计中，选择树莓派专用的操作系统 rasbian 来作为开发的操作系统。选择树莓派的原因有以下几个：（1）首先，树莓派在保持体积小，可以进行嵌入式开发的前提下，具有所需的网络外设功能。基于 linux 的树莓派可以通

过 python 和 C 语言实现快速的网络设计开发。从本质上来看，它和一般的基于 linux 的主机或者基于 linux 的服务器没有区别。同时，通过外接 usb 无线网卡，树莓派可以作为物联网设备接入环境中的无线网络。除此之外，使用 usb 无线网卡的树莓派也可以在对应的无线网卡端口建立无线网络，让树莓派自己成为无线网络的接入点。（2）其次，在前面已经提到，树莓派的操作系统和数据文件装载在 SD 卡中。这也就意味着，当拥有一套成熟的能够实现物联网功能的设备操作系统备份时，就可以通过烧写这些操作系统和文件系统的镜像，快速的制造大量物联网设备。而如果使用传统的嵌入式开发平台，物联网设备的安装和配置将会耗费大量的时间。（3）树莓派的维护和维修是非常方便的。树莓派可以通过 usb 标准接口充电，因此在电源上可以非常方便的实现。同时对于操作系统数据和无线网卡的更换可以通过替换 SD 卡和 usb 外设来快速实现。在这点上，树莓派优于其他的嵌入式框架。

在搭建物联网平台时，设计中选用 Realtek 瑞昱开发的 RTL8188CUS 无线网卡 usb 作为网卡外设。树莓派最新发行版本是自带该无线网卡驱动的，但是在尝试使用该无线网卡开启无线 WiFi 时，遇到了一些问题。构建无线 WiFi 时使用了 hostapd。hostapd 是一个在 linux 操作系统中的用于打开 AP（无线 Access Point, AP）的软件。它可以实现用户态下的无线网络相关接入管理。然而标准的 hostpad 是不支持 RTL8188CUS 的驱动的。因此需要通过交叉编译的方法，修改 hostapd 程序的源代码，从而编译一个支持 RTL8188CUS 驱动的 hostapd 软件开启无线 WiFi。

3.2.2 网络二三层设计

毫无疑问，Ad Hoc 和基础架构模式各有千秋。因此，在对它们进行选择的时候，需要考虑具体的运用场景。

在智能电表物联网中，电表在竖井中的状态相对稳定。电表所放置的位置是相对稳定的，电表之间的相对位置也因此是相对稳定的。在竖井中，信道的变化也相对比较稳定。因此，电表之间的连接不会出现大的变化。在这种情况下，基础架构模式能够提供更加稳定的信号传输，而 Ad Hoc 模式的几个主要的面对结点快速变化场景的优点都无法表现出来。

其次，在智能电表无线物联网设计中，安全性是一个重要的考量。在上文中已经提到，物联网设备在安全性上更容易受到攻击，而 Ad Hoc 网络又有着明显的安全性的不足。而于此同时，基础架构模式可以对结点的接入做到更加好的控

制。在基础架构模式中，系统可以在包括防火墙，接入设备监控认证，防止恶意攻击等一系列安全保护上做到更方便有效的配置。

一个竖井中电表的数量也相对较多。在上文中我们提到，随着连接设备数量的增多，Ad Hoc 的性能开始显著下降。在[11]中，超过 10 个物联网设备时，性能就已经出现了明显的下降。而相比较，基础架构模式可以同时提供良好信号传输的设备数量，因而可以满足一个竖井中电表的数量情况。

功耗也是另外一个重要的考量。实际情况中，竖井智能表放置位置相对简单线性，同时智能表之间也没有通信需求。Ad Hoc 模式引入了多余的路由转发过程。而对于物联网设备来说，智能表的电量和功率可能都是非常有限的，因此使用基础架构模式更加合适。

综合以上几个运用场景和上文提到的 Ad Hoc 模式和基础架构模式的优缺点，本项目设计选用基础架构模式而不是 Ad Hoc 模式来作为设计的无线物联网的基本网络模式。

同时在上文的分析中可以看到了 IPv6 在物联网中的巨大优势，因此在设计中，系统选择基于 IPv6 的无线物联网架构。

3.2.3 物联网网络结构设计

在决定了底层的设计之后，就可以根据这个设计搭建无线物联网。图 3.1 是网络结构拓扑图。可以看到，物联网结构由以下几个关键部分组成。

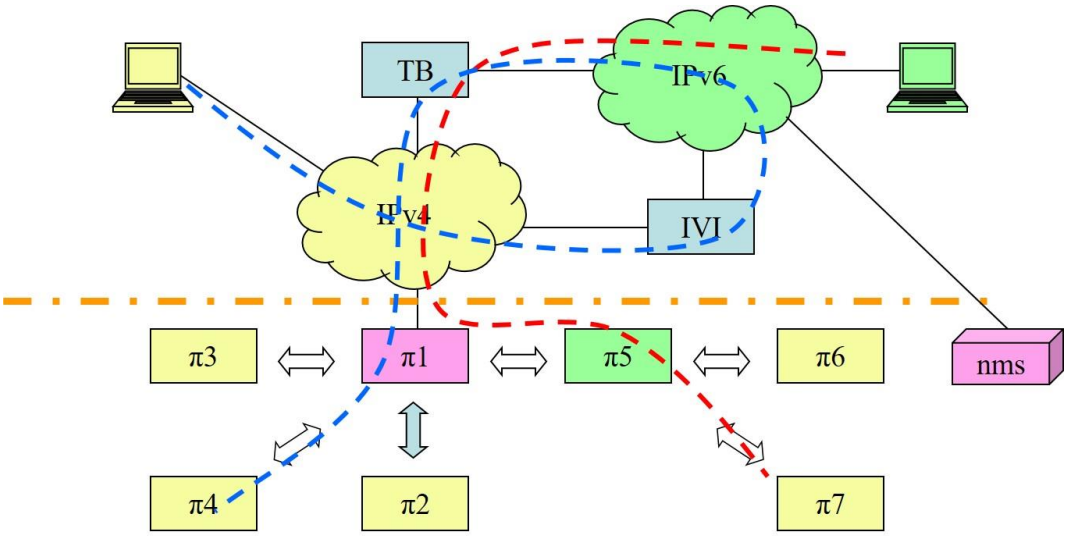

图3.1 无线物联网络设计拓扑图

第一个部分是物联网 WLAN 的 Access Point 设备，在下文中为了表达的简洁明了，将其简称为 AP。在图中，它被表示为 π1。在智能电表的每一个竖井中，都会有至少一个这样的 AP。在下游的无限物联网中，所有的树莓派物联网设备（π2，3，4，6，7）统一由 AP 进行管理。管理的内容包括无线网络接入，地址分配。经由 AP 的路由，所有的物联网普通设备从而可以获得对外的网络连接。

在上游，AP 向地址服务器请求需要的子网前缀地址和 IVI 地址空间，在获得可以使用的子网前缀和 IVI 地址之后，下游的物联网设备才可以获得可以使用的地址。AP 上游是一个基于 IPv4 的以太网环境，这是考虑到很多地方的网络环境一般不支持 IPv6。因此这个时候，直接路由下游物联网设备的 IPv6 数据包是不能成功的。因此 AP 作为隧道的客户向隧道服务器发起隧道请求，将 IPv6 数据包封装到 IPv4 数据包之中。数据包在隧道服务器端被还原为 IPv6 包发送至 IPv6 外网。这样就实现了物联网设备的 IPv6 的外网连通性。具体的设计在下文中进行阐述

第二个部分就是物联网设备（π2，3，4，6，7）。在后文为了表达的简洁，同样，本文把它们简称为 Client。如上面所说，物联网设备需要向 AP 申请网络连接，再经由 AP 路由和隧道封装完成与外网的对话。

Client 本身作为物联网的设备，需要对环境中的数据进行交互。在研究的运用场景中，智能电表就是对应的 Client。智能电表需要搜集环境中的数据。并且向上对外网提供一个方便的数据接口。

第三部分则是中继 AP。在竖井中，末端的智能电表可能距离 AP 太远。虽然根据以往的研究[22]，基于 WiFi 的 AP 可以实现公里级别的无线连接。但是考虑到民用场景低功率的情景，预计 20m-100m 外的信道会开始显著恶化。因此在竖井中，设计框架考虑了中继 AP 的情况。同时值得注意的是，比起单纯的中继 AP，在这里，设计框架中考虑一种更加普适的情况。可以注意到，一个 Client 扩展 AP 中继功能的过程中，并不会引入过多的计算和传输负担。因此，在设计中将部分 Client 的功能进行扩展，形成一种新的物联网设备：AP/Client。它既作为 Client，也作为一个子 AP。在后文中，本文统一称呼它为 AP/Client。

第四部分是网络管理服务器（network management system, NMS）。通过这个 NMS，管理者可以监控下联物联网设备的状况，并且对这些物联网设备进行访问。

第五部分是 IVI 翻译服务器。外网在只有 IPv4 支持的情况，通过设备的 IVI 地址对应的 IPv4 地址和端口号，由 IVI 翻译服务器进行翻译。这样就可以实现所设计的无线物联网中 IPv4-IPv6 的过渡和兼容。

为了方便理解，可以简单的将设计框架进行分层抽象。AP 作为物联网设备和外网通信的管理者，可以抽象为“接入层”。Client 可以抽象为“设备层”。在物联网设计中的隧道服务器，地址服务器，IVI 服务器主要服务通信连接，因此可以抽象为“通信层”。

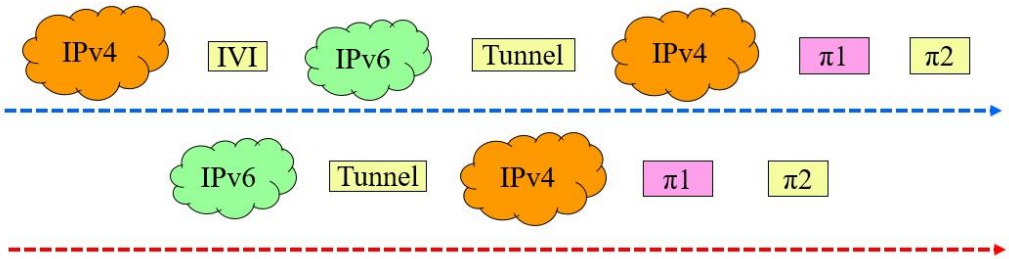

图3.2 IPv4和IPv6发起的对话示意图

在外网尝试与物联网设备建立通信的时候，有以下两种方式：（1）外网处于 IPv6 环境，直接对物联网设备发起对话。此时的 IPv6 数据包被路由至隧道服务器端，被隧道服务器封装。封装后经由 IPv4 网络环境发往对应竖井的 AP。AP 收到数据后将数据包还原，路由或者转发至对应物联网设备。物联网与外网的通信则是将这个过程反过来。（2）外网处于纯 IPv4 环境。这个时候，外网将数据包发往 IVI 翻译服务器。数据包发往的 IPv4 地址和端口号将由 IVI 翻译服务器翻译成对应的 IVI 型 IPv6 地址。于是此时 IPv4 数据变为一个普通的 IPv6 数据包被发往隧道服务器。在这之后，隧道服务器使用和（1）中一样的方法对数据包进行封装。

3.3 6over4 隧道的基本原理

如上面所说，在大多数情况下，可以使用的网络主干是不支持 IPv6 的。因此，系统需要在 AP 的以太网接口开启隧道服务，将数据包进行封装。系统使用基于 OpenVPN 的 6over4 隧道技术[23]。OpenVPN 是一种在 linux 操作系统上非常常见

的虚拟网络技术。在 OpenVPN 中，通过虚拟设备可以实现数据的点对点传输。在具体设计中，网络通过 OpenVPN 隧道，将 IPv6 数据包通过 IPv4 网络传输，实现物联网 IPv6 设备和外网 IPv6 的端对端通信。在实际的设计中，隧道包括了隧道代理服务器，隧道服务器和隧道客户几个部分。

客户端通过 IPv4 向隧道代理服务器发出建立隧道服务的申请。隧道代理服务器在接受申请后，对客户端进行用户认证，并给客户端反馈对应的隧道服务器信息。客户端在获得隧道服务器信息后，修改自身 OpenVPN 的配置信息，开始向由代理服务器分配的隧道服务器发起隧道建立请求。隧道的基本框架使用了本实验室开发的[24]隧道框架，通信的安全可以得到保障。具体的配置细节可以在这篇文章中查看。

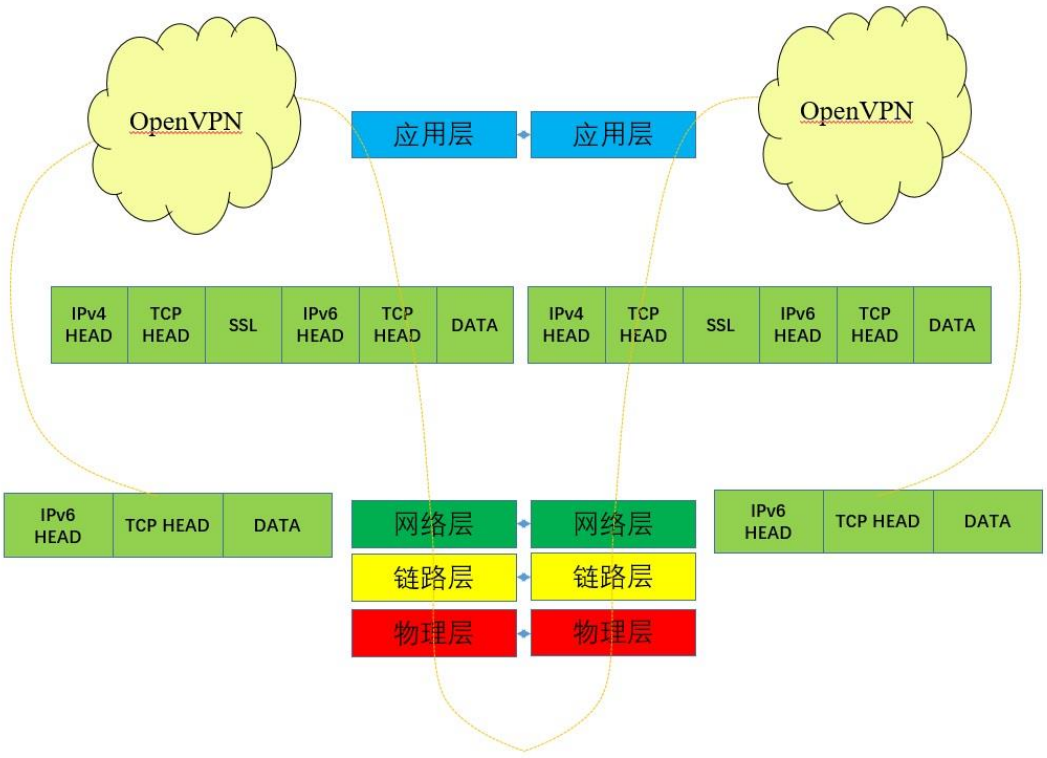

图3.3 基于OpenVPN的隧道链接示意图

客户端设备在本机上建立和 6over4 隧道服务器相对应的虚拟端口。所有的 IPv6 数据包被该虚拟端口截获，送往在应用层上的 OpenVPN 程序。在 OpenVPN 程序中，IPv6 的数据在会话层加入 SSL 确保数据加密之后被进一步封装到基于 I

Pv4 的 TCP/UDP 数据包之中。封装好的 IPv4 数据包这个时候才被真正的物理接口发往隧道的另一端，也就是服务器端。从网络层接口的角度上来看，数据包似乎通过虚拟接口直接发送到了隧道另一端的 IPv6 网段。这就是 6over4 隧道服务的基本概念。基于 OpenVPN 的 6over4 隧道服务数据传输示意图如图 4.1。可以看到，OpenVPN 给我们提供了很好的封装接口。

3.4 IVI 地址的基本原理

对于现阶段的网络结构来说，主体仍然是 IPv4 的。因此 IPv4/IPv6 的转换技术非常的重要。IVI 翻译是 IPv4 到 IPv6 地址转换的一种技术。通过 IVI 技术，IPv4 和 IPv6 的用户可以保持端对端的通话和地址透明。本质上，它是一种无状态的 46 网络地址转换机制（Stateless NAT64）[25]。IVI 地址翻译的详细机制不是本次毕设的重点，系统中使用的主要设计来自于实验室师兄盛家茂学长提供的 IVI 模块[26]。简单的思路可以介绍如下：IVI 技术是为了保证 IPv4 和 IPv6 平稳过渡而诞生的技术之一。和其他的 46 过渡技术想比，本项目使用的 IVI 技术的一个重要特征就是，考虑到 IPv4 公网地址的稀少。因此在使用的 IVI 技术中，IPv4 公网并不是与 IPv6 公网地址进行一对一的绑定。IPv4 公网地址实际上是与 IPv6 的前缀进行了一对一的绑定。在这种情况下，实际 IPv4 和 IPv6 地址的绑定是 1：N 的对应关系。当用户在 IPv4 的网段尝试对一个 IPv6 网段的 IVI 地址发起对话时，用户向这个 IVI 类型 IPv6 地址对应的 IPv4 和端口号发起 HTTP 请求，IVI 服务器就会自动的将 IPv4 地址和端口号进行翻译，将数据包进行拆分重组，变成 IPv6 数据包发往 IPv6 网络环境。

在智能电表无线物联网中，每一个电表设备都被分配一个 IVI 地址，比如，一个 Client 设备可能被分配 2001:da8:e300:79c2:a889:400d::/96 这个 IPv6 的地址。而 2001:da8:e300:79c2:a889::开头的前缀，对应的是 121.194.168.137 这个 IPv4 地址。系统使用 1：16 的 IVI 复用比率，对应的映射关系可以简单的表达如下：一个设备的地址的端口可以通过下面的公式计算：1024 + mod(2001:da8:e300:79c2:a889:400x - 2001:da8:e300:79c2:a889:4000, 16)。比如 2001:da8:e300:79c2:a889:400d，在上面的计算中会得到 mod(d, 16)，结果就是 1037 端口。因此，对这个地址进行基于应用层的控制，用户只需要在 IPv4 网段对 121.194.168.137:1037 地址和端发送请求就可以做到了

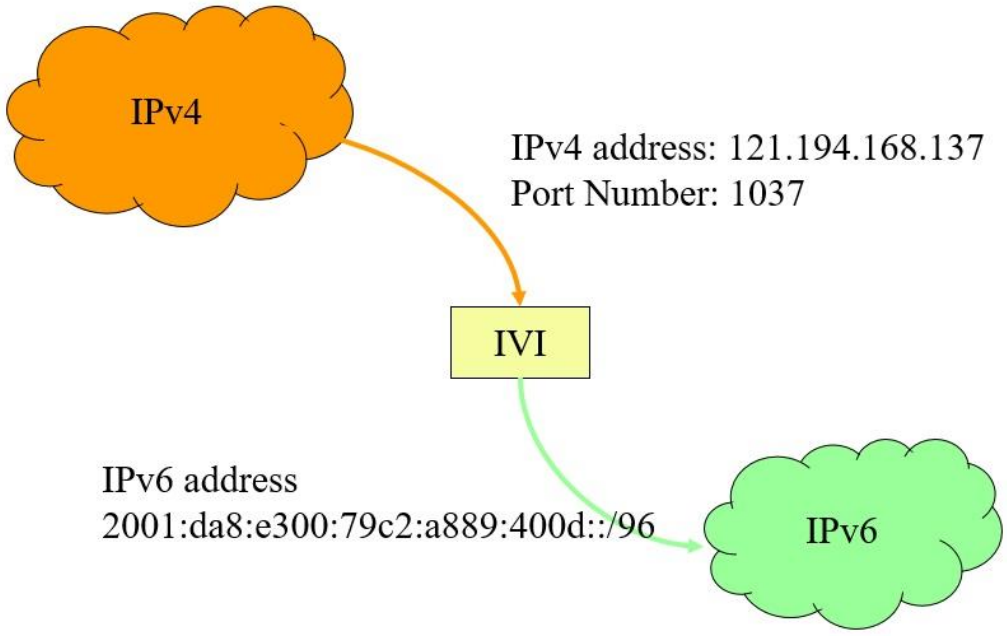

图3.4 IVI地址翻译示意图

第4章　无线物联网设计细节

4.1 AP 物联网设备配置

AP 负责在下游（1）物理上建立 WLAN。（2）逻辑上管理连接 WLAN 设备的地址分配和数据包转发。在上游则负责与通信层进行连接。下面，按照 AP 工作时的时间顺序，对 AP 的具体设计进行阐述。

4.1.1　建立隧道连接

AP 在开机的同时，开始通过以太网接口连接至外网。在以太网可以连接至 IPv4 外网之后，AP 向隧道代理服务器发出服务申请。之后按照上文中所描述的方式，AP 向所分配的服务器发起 6over4 隧道建立请求。

在获取了 6over4 隧道服务器信息后，AP 设备在本机上建立虚拟的 tap0 端口。从而和隧道服务器建立隧道连接。

4.1.2　获取与管理地址空间

在 AP 建立隧道的同时，AP 需要向地址服务器发起地址请求。请求的内容包括可以使用的地址前缀和 IVI 地址。可以使用的地址前缀为 IPv6 地址前缀，AP 使用一个地址前缀并将自己配置为该子网的网关，从而开启 WLAN 服务。而其余的前缀，则可以在收到 AP/Client 的地址前缀请求时分配给下联的 AP/Client 建立子网 WLAN。在获取地址的同时，地址服务器将对应的路由 IPv6 信息告知对应的隧道服务器，这样整个网络的路由就成功的建立起来了。在实际运用中，地址服务器可以和隧道代理服务器放在一台物理机器上实现。

对于 AP 来说，需要管理的地址主要包括全局 IPv6 地址前缀，物联网设备全局 IPv6 地址，和物联网设备的 IVI 类型 IPv6 地址。在下面的章节中，本文会讲解 AP 如何管理和分配下联物联网 Client 设备的全局 IPv6 地址。IVI 类型的 IPv6 地址则使用了一个新的协议进行管理，会在第五章中进行详细讲解。值得注意的是，为了做到方便易于维护的地址获取，AP 上搭建了一个基于 Django 的数据服务器。同时，为了体现封装性，AP 使用了 Django 内置的数据库来进行数据管理，而不是分裂的使用现有数据库接口。下属 Client 通过对 AP 上搭建的 Django 服务器发起应用层请求来与 AP 上的数据库进行交互。在 AP 的全局 IPv6 访问

上，为了设计的剪接性，物联网设备统一使用 70 端口作为 Apache 的监听端口。在这种情况下，AP 的数据服务器可以通过地址和端口进行访问：举个例子，当 AP 的 IPv6 全局地址为 2001:da8:b100:101d::1 的时候，物联网设备 Client 或者 AP/Client 只需要通过下面的地址 http://[2001:da8:b100:101d::1]:70/就可以和 AP 的 HTTP 数据信息交换。值得注意的是，Apache 可以监听任意 AP 上拥有的地址对应的端口 HTTP 请求。在访问 IVI 地址的时候，端口是受限的，这一点在下面会提到。

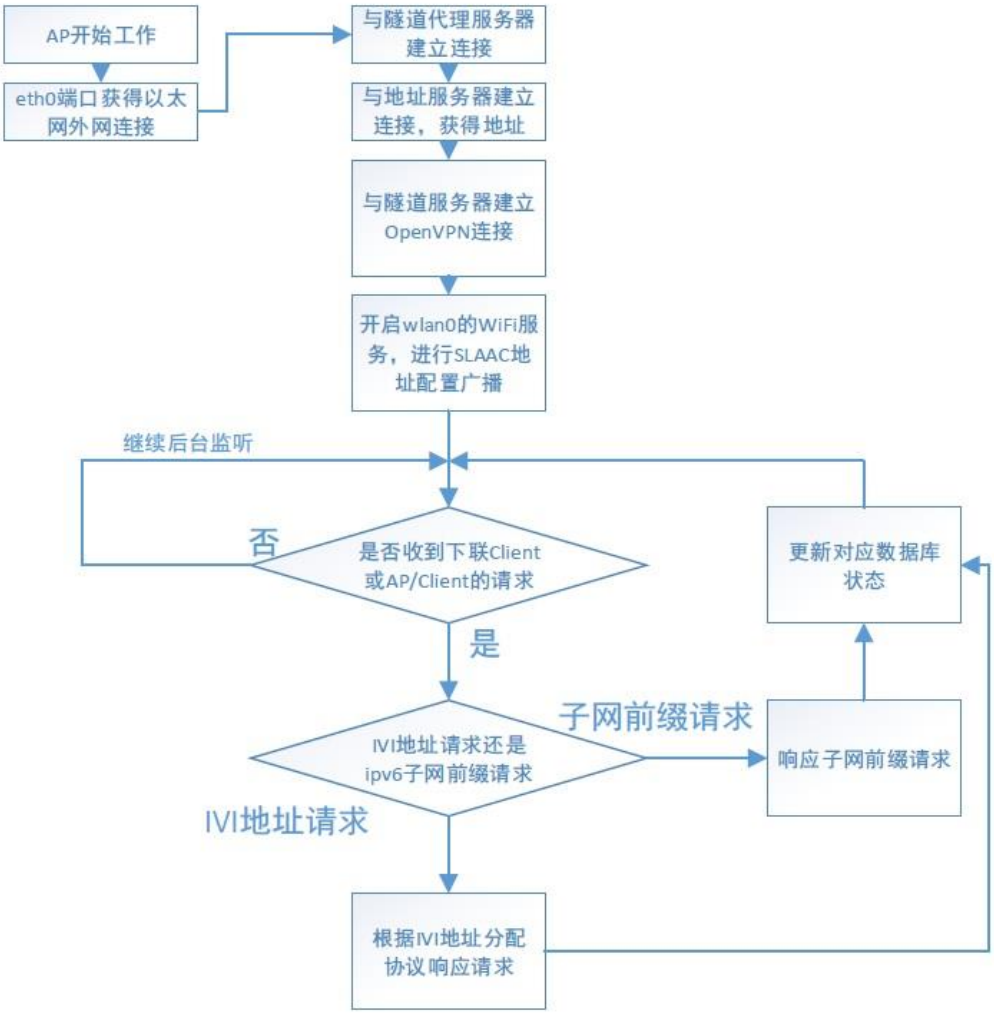

图4.1 AP设备工作流程示意图

4.1.3 开启 WLAN 服务

AP 在获取全局 IPv6 前缀之后，就可以作为 Access Point 开启对下游物联网设备的 WLAN 服务。在无线物联网中，安全性是非常重要的考量因素。以往常使用的有线等效保密（WEP）协议被证明可以被第三者通过截获和伪造冗余信息的方法破解[27]。因此设计中的 WiFi 使用更为安全的 WPA2（Wi-Fi Protected Access）协议。同样在文献[27]中可以看到，现阶段基于 WPA2 的 WLAN 是相对安全的，没有严重的安全漏洞。AP 使用 hostapd 在 AP 的无线接口开启 WiFi 连接。WiFi 的密码，加密方式和 SSID（服务集标志）是保证物联网设备合法访问的防护。

图4.2 生成SLAAC IPv6地址的基本流程

在地址管理上，为了节省 AP 的资源和减少配置上线的时间，SLAAC （Stateless address autoconfiguration，无状态地址自动配置）被应用作为地址配置机制。在 AP 拿到可用的子网前缀之后，AP 选取一个空闲的子网前缀，使用 radvd 并且修改 radvd 的配置文件在 WLAN 中进行 SLAAC 64 位 IPv6 地址前缀的广播。SLAAC 地址生成的机制如下[28]：

每一个接入 AP 的 WLAN 的物联网设备需要根据自己的无线网卡 MAC 地址来生成自己的 Modified EUI-64 标识符。可以通过以下几个步骤来进行生成：（1）读取自己的无线网卡 MAC 地址。无线网卡 MAC 地址在出厂时被终身绑定，是

一个 48 比特的地址。（2）生成 64 比特的 EUI-64 地址。这一步是通过在 24 比特处加入“FFFE”对应的 16 比特来实现的。（3）考虑到 IPv6 中，存在一个 Universal/Local（全局本地）比特标志，因此需要将 EUI-64 地址中的第 7 位转换为 1。这代表这个地址是全局地址。另外，最新的 SLAAC 中，为了进一步防止地址被追踪，也存在和上面的不同的地址随机生成方法。

在实际使用 SLAAC 地址配置的时候，物联网 AP 设备选取一个可用的前缀，比如 2001:da8:b100:101d::/64，并且在无线端口（比如 wlan0）将自己配置成网关 2001:da8:b100:101d::1/64。然后，它开始在无线端口开始广播可用前缀 2001:da8:b100:101d::/64 的信息。而已经接入了无线网的物联网 Client 设备，这个时候收到前缀信息，就开始使用这个前缀对自己进行自动的无状态地址配置。Client 设备读取自己的无线网卡 MAC 地址，根据上一段描述的方法生成可用的 Modified EUI-64 标志，和前缀进行结合，生成 128 比特 IPv6 全局地址。通过以上几步，任何一个合法接入 AP 的物联网 Client 设备都可以成功的对自己进行全局 IPv6 地址的自动配置。更详细的协议细节，参见[28]。

4.2 Client 物联网设备配置

可以看到，Client 和 AP 的联系是非常紧密的。这一章节从 Client 的角度继续描述物联网设备配置。

4.2.1 Client 工作流程设计

和 AP 的交互是 Client 物联网设备正常工作的前提。在上一章节中已经描述了合法接入物联网的 Client 设备如何配置自己的全局 IPv6 地址。对于物联网设备来说，它们还需要考虑自己的 IVI 地址获取。

在 Client 设备开机之后，Client 会首先尝试连接 AP 或者 AP/Client 设备提供的 WiFi。这一点是通过在 Client 内部写入合法的 SSID，加密方式和密码实现的。在 raspbian 操作系统中，和 debian 系 linux 操作系统类似，用户可以配置 wpa-supplicant.conf 文件。通过在这个文件中写入所有可能的合法 SSID 和密码组合，并且在 raspbian 操作系统网络接口配置文件/etc/network/interfaces 中建立索引，就可以实现 WiFi 的自动主动连接。同时，对应的配置文件通过 root 权限进行修改和读取，这也意味着用户不需要担心密码和 SSID 泄露的问题。

在合法的连入无线物联网之后，通过上一章节的方法，物联网 Client 可以自动配置自己的全局 IPv6 地址。在获得 IPv6 地址之后，Client 再通过应用层协议申请 IVI 地址。通过以上的设计，Client 设备就成功的获得了 IPv6 和 IPv4 与外网的连通性。

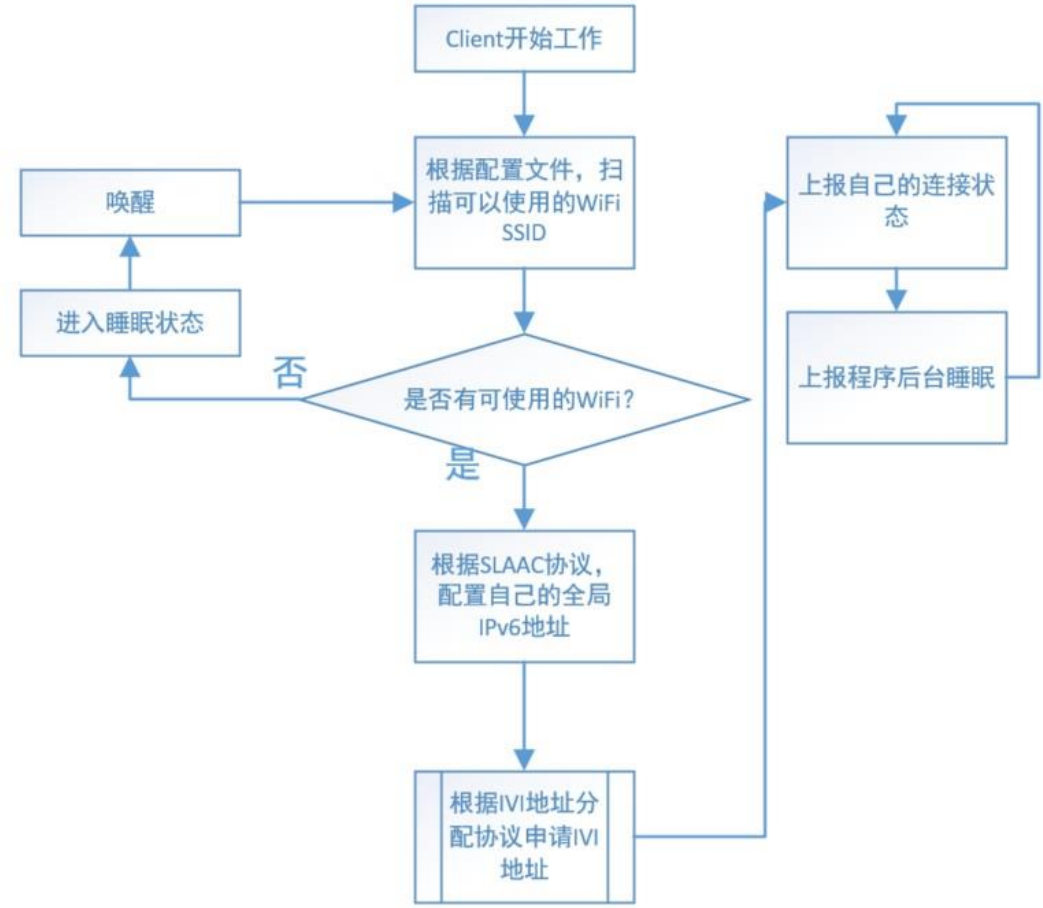

图4.3 Client设备地址配置工作流程示意图

4.2.2 Web 服务器

在前面已经提到过，Client 在具体运用场景中就是智能电表。因此数据的搜集，保存和传输是非常重要的。

在设计中，根据 Django 框架，在 Client 上上线了一个数据服务器。一方面，通过 python 脚本经由 localhost 地址对自身数据服务器进行访问，在 Client 上设备可以实现数据的连续采集。另一方面，通过数据服务器，Client 对外提供包括实

时数据读取，历史数据读取在内的数据读取接口。考虑到现有的树莓派暂时还是智能电表的开发演示，因此 Client 并没有读取真实数据的能力。因此 Client 模拟电表的真实数据，同时提供一个简洁的数据写入接口。这样，当转换到真实的智能电表上时，通过对接口进行操作，Client 可以迅速的实现真实数据写入功能。注意和 AP 类似，Client 上的数据服务器同样使用基于 Apache 和 Django 的服务器架构，访问的方式也保持了一致。如果 Client 设备的地址为：2001:da8:b100:101d:8024:d511:bbcd:6b7d，那么用户可以通过 http://[2001:da8:b100:101d:8024:d511:bbcd:6b7d]:70/下的地址来进行访问。在现在的设计中，外网用户可以通过下面的地址 http://[2001:da8:b100:101d:8024:d511:bbcd:6b7d]:70/serve/now/和 http://[2001:da8:b100:101d:8024:d511:bbcd:6b7d]:70/serve/short_term/来分别请求实时数据或者上一批数据。之所以提供批量的数据请求接口是因为，建立 HTTP 连接以及建立 SSL 加密的 HTTP 连接是需要额外的时间开销的。因此，如果能够一次性批量请求数据，可以减少相对的额外损耗。此外，为了方便管理 Client 的参数，Client 还提供了 http://[2001:da8:b100:101d:8024:d511:bbcd:6b7d]:70/change_settings/接口来对 Client 的各种系统参数进行调整。在上面的所有 HTTP 请求中，数据传输都可以通过 SSL 协议来对数据进行保护，防止监听，保护数据安全。同时，通过使用公钥私钥的加密方式，每次物联网设备间的通信需要出示密码。这样的话，就可以保证接口的安全不被外网非授权设备访问和破坏。这一点在其他的设备，包括 AP，AP/Client，NMS 上的服务器都是同样的。

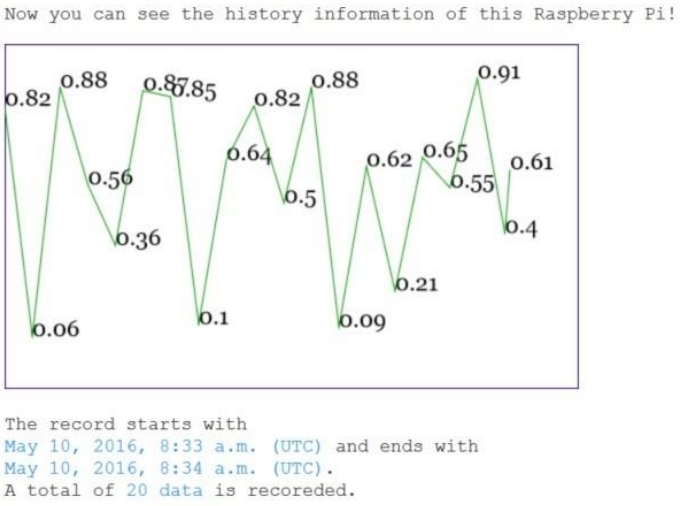

图4.4 Client设备数据搜集结果示意图

4.2.3 连接状态汇报

在物联网搭建中，一个重要的问题是，如何判断一个设备的连接状况。在传统的物联网地址分配中，无论是 DHCPv6 这样的有状态分配还是 SLAAC 这样的无状态分配协议，都无法对连接状态进行检测。静态地址配置也会有同样的问题。下联的物联网设备可能因为电量耗尽，信道恶化，无线网卡或者操作系统损坏而丢失连接。因此，设计时需要在 Client 端设置合适的连接状态上报机制，在 AP 和 NMS 设置对应的状态更新机制。

对于 AP 来说，它需要搜集对应物联网设备的地址配置情况。当一个设备失效之后，分配出去的地址应当被回收。因此，Client 需要对 AP 间隔性的汇报自己是否存活。对于 NMS，需要 Client 上报的东西就更加复杂了。因为 NMS 作为网络管理系统，不仅需要监控 Client 的连接状况，还需求搜集 Client 包括地址信息，MAC 地址等一下信息来进行管理。

因此，Client 首先需要对 AP 进行“存活反馈”。在设计中，Client 定时的对 AP 上的 Django 服务器发出 HTTP 数据包。在设计中，“/heart/”是 Clien 的存活反馈相对网址接口。

AP 在收到数据包之后，更新下联 Client 设备的连接状态。当一个 Client 超时没有回报之后，将被 AP 回收地址，标记为下线。对 NMS，则需要通过 HTTP 中的 POST 方法不断的向 NMS 上传最新的信息数据。事实上，包括 AP 在内的所有物联网设备都需要定期向 NMS 汇报自己的最新信息。在现阶段的设计中，上传的数据包括：（1）设备的类型。可能的类型包括：“AP”，“AP/Client”和“Client”。（2）AP 子网前缀。对于 AP 和 AP/Client 来说，它们需要汇报自己开启的 WiFi 使用的 SLAAC 协议广播的前缀是什么。（3）无线网卡 MAC 地址。（3）全局 IPv6 地址。值得注意的是，AP/Client 可能会有多个不同子网的全局 IPv6 地址。考虑到每一个 IPv6 地址都可以使用，虽然 NMS 会搜集所有的 IPv6 地址，但是在网页上只会显示一个地址。（4）IVI 类型的 IPv6 地址。

在设计中，物联网设备使用 python 上的 url 库来实现上报功能。在后台运行的 python 程序每次完成信息的上传，就会进入睡眠。睡眠时间结束后进行下一次的信息上传，周此往复。睡眠的时间是 Client 的参数，可以通过上面提到的 HTTP 接口进行设置。

4.3 AP/Client 物联网设备配置

在上文中提到，为了保证通信的流程，延伸通信的距离，系统设计中引入了AP/Client 这种类型的物联网设备。首先，它具有普通 Client 设备所有 Client 功能。它使用相同的方式向 AP 请求 IPv6 地址和 IVI 地址，需要向 NMS 上报所有的信息，也搭载着一个数据服务器。它多出的功能在于它能够使用一个新的子网前缀提供 WiFi 接入。

在 AP/Client 中存在两个无线网卡，因此在操作系统中可以看到两个无线接口 wlan0 和 wlan1。wlan0 负责连入 AP 提供的 WiFi，并且成为 AP 管理子网的一部分。wlan0 上的 IPv6 全局地址和 AP 一致。比如，AP 的子网前缀为 2001:da8:b100:101d::1/64，拥有网关 2001:da8:b100:101d::1/64 的情况下，wlan0 的地址可能为 2001:da8:b100:101d:8024:d511:bbcd:6b7d。在和普通 Client 一样获得通讯地址之后，AP/Client 向 AP 的:70/prefix/的 url 地址发出前缀请求。可以看到前缀请求同样是基于应用层的 HTTP 请求。当 AP 收到这个请求后，如果判断请求合法，就会从自己的数据库中寻找可用的未分配 IPv6 前缀。如果存在可用的 IPv6 前缀，就通过 HTTP response 返还给 AP/Client，同时 AP 需要修改自己的路由表。收到地址的 AP/Client 于是开始对自己的 wlan1 进行配置。在这个角度看，对于 AP/Client 来说，wlan0 端口实现的是 Client 的功能，而 wlan1 则是实现的 AP 的功能。

在设计中对于 Client 来说，连接到 AP 或者 AP/Client 对它来说都是等价的。在路由的层面上，AP 需要对前缀的分配状态进行监控，从而保证路由的通畅。

实际上，在设计子网选择的时候，系统也可以使用“网桥”设计。在这种情况下，wlan0 和 wlan1 在逻辑上是相连的。这样，来自 AP 的 SLAAC 协议包也会被在 wlan1 进行转发。这样只需要一个子网前缀就可以完成一个竖井中所有的设备配置。整体上来看这样的设计更加整洁。但是这样做对于系统的调试带来了困难。无线物联网的一大特点就是设备可能随时会发生断电，设备损坏和通信中断。在这个时候，前缀实际上可以蕴含物联网的层次和结构信息。在调试和维护的时候，系统就可以使用子网前缀作为调试信息。而使用同一前缀的物联网络，需要使用额外的存储空间和运算来记录结构信息。

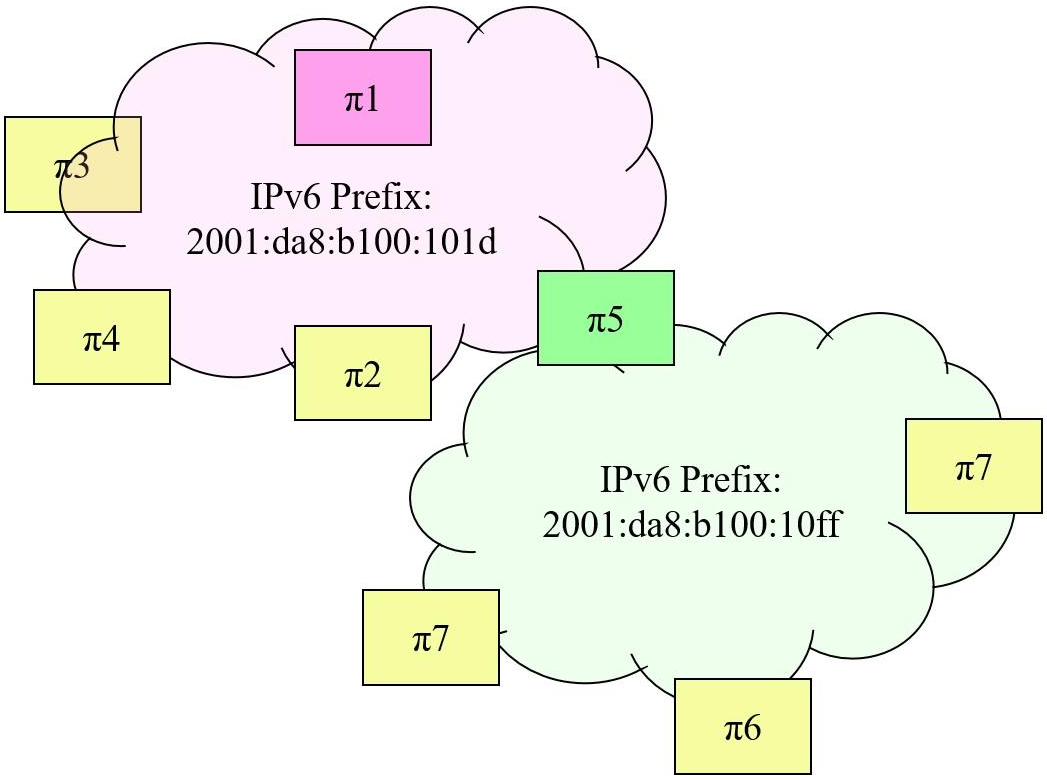

图4.5 AP/Client与AP子网前缀拓扑图

4.4 NMS 配置

在上文中已经介绍了 AP，Client 和 AP/Client 向 NMS 汇报数据的主要内容和原理。下面介绍 NMS 的网站结构和设计。在设计中，一台树莓派机器作为 NMS 被分配了 IPv4 和 IPv6 地址，这样无论通过哪个 ip 协议用户都可以对 NMS 进行实时的访问。现在的实验配置中 NMS 设置了一个公网 IPv4 地址 121.194.1617.98 和公网 IPv6 地址 2001:250:3::191。通过这两个地址，用户可以访问 NMS 的控制服务器网页。

在 NMS 中，为了方便对智能电表物联网设备信息的管理，网页使用了 Javas cript 和 CSS 对界面进行了美化。界面使用了 purecss 来进行美化。在 purecss 中，提供了很多格式美观的表格，按钮控件。

在 NMS 中，用户可以简洁的看到关于 AP，Client，AP/Client 各种信息。如上面所说，可以观察到的信息还包括：全局 IPv6 地址，IVI 地址，下联和上联的设备情况。在 NMS 中还需要在后台实现几个功能。（1）物联网拓扑结构的计算。

这里就可以和上面的前缀分配方式联系起来了。因为 AP/Client 使用的不同的子网，因此 NMS 可以直接通过前缀建立拓扑图。这一点是在使用同一子网无法快速实现的。对于一个设备，NMS 首先查看它的前缀 prefix 数据，如果是 AP 或者 AP/Client，NMS 就可以根据它的 prefix 数据（Client 用户在这一项是 None）对其他的设备进行匹配。匹配上的设备，就会被标为下联设备（在网页中显示为 Current Connecting Pi）。同样的道理，NMS 也可以根据 Client 和 AP/Client 自身的 wlan0 端口的全局 IPv6 地址，判断它们属于哪个 AP 下联。

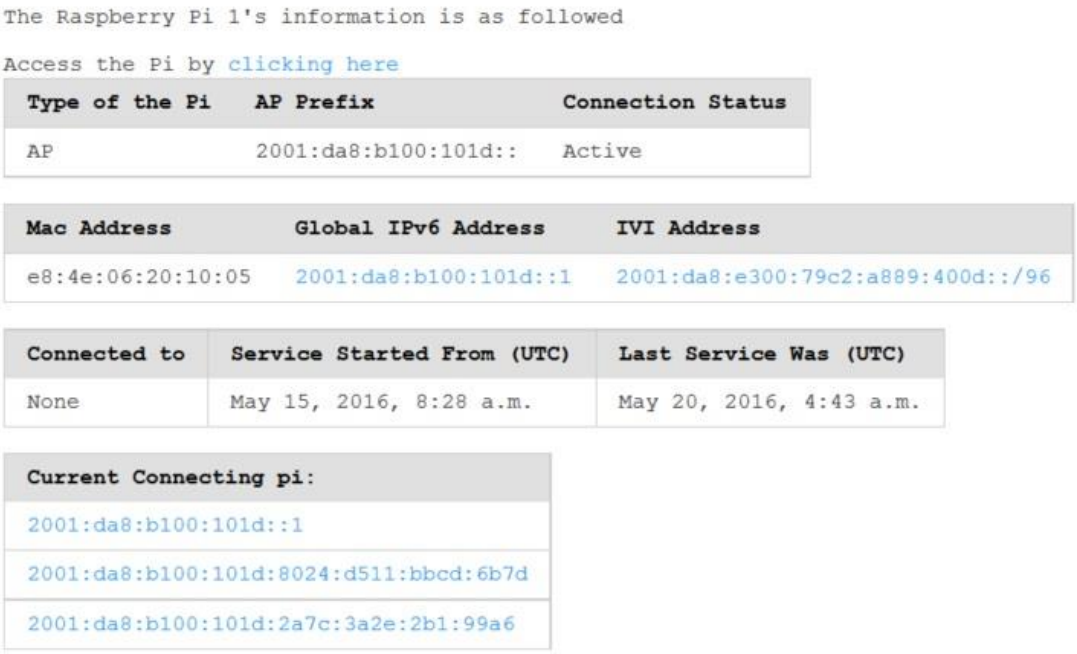

图4.6 NMS控制界面效果图

第5章　IVI 地址分配机制

5.1 IVI 地址分配机制

在问题研究中，设备和普通的主机不一样。一方面，设备可能断电，设备也可能因为通信不畅，短暂的失去和 AP 的通信连接。同时，设备可能会发生损坏。树莓派的 SD 卡可能发生物理性的损坏。无线网卡在树莓派设备中可能需要工作非常长的时间，而无线网卡的寿命是有限的。环境中的不可控因素，比如潮湿的环境也可能会影响无线网卡的正常工作。

在这样的前提下，设计系统时就需要考虑一个问题，如何确保物联网设备在重新连接之后，保持地址的一致性呢？某一款应用程序可能和一个物联网设备进行了交互，并且为了下一次交互的需要，它记载了目标物联网设备的地址。因此如果地址发生了变化，物联网的功能会遇到很多的问题。更加糟糕的是，这个地址可能已经被分配给了别的完全不相干的新连接的物联网设备。这样物联网就不仅会发生传输失败，还会发生传输错误。

因此，在 IVI 地址分配的时候，系统使用了一个基于应用层 HTTP 协议的 IVI 地址分配协议。之所以使用基于应用层 HTTP 协议是因为，AP，Client 上都已经搭好了使用 Apache 的 Django 服务器。因此在 HTTP 上设计的地址分配协议，无论是开发的简洁性，可读性，还是协议的安全性，都存在优势。

在 AP 中，IVI 地址的数据库管理发生以下的变化，对于每一个 IVI 数据类型，在数据库中定义它包含四个数据子类型：（1）IVI 地址。这是 IVI 实际的 IPv6 地址，比如 2001:da8:e300:79c2:a889:400d::/96。（2）MAC 地址。对于已经被使用过或者曾经被使用过的地址，AP 会记录使用这个地址的树莓派（或者其他设备的）的无线接口的 MAC 地址。（3）PID。这是一个树莓派所绑定的编号，下面会详细说明。（4）地址状态。地址状态可能有“未使用”（或者“可使用”），“已分配”和“已使用”三种。

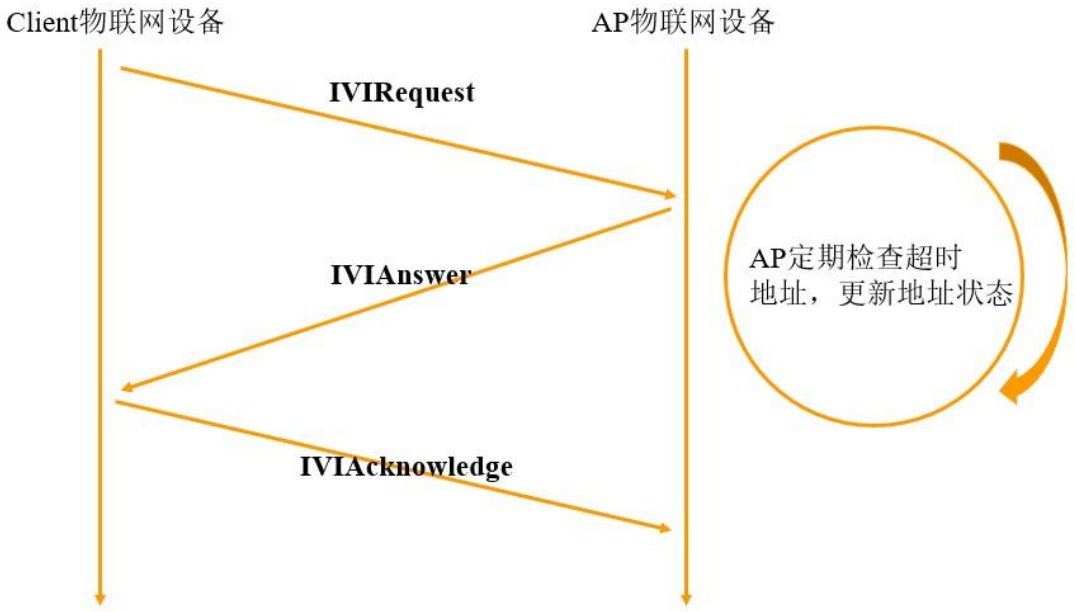

图5.1 IVI地址分配协议示意图

5.2.1 地址分配机制

系统设计的 IVI 地址分配协议分为三步：（1）IVIRequest。在这个部分中，Client 设备向 AP 设备的数据服务器发送地址申请请求。在这个请求中，Client 同时上传自己的 MAC 地址和 PID。AP 在收到请求后，首先查找对应 MAC 地址和 PID 的使用记录，如果有匹配，立马选择对应的 IVI 地址。如果没有匹配，则从“未使用”的 IVI 地址中选取一个地址。在选取地址之后，AP 将对应的 IVI 地址状态更新为“已分配”，并且更新对应的 MAC 地址和 PID 信息。同时修改路由表项目，将 IVI 地址的路由表进行建立。值得注意的是，如果 AP 发现，发出请求的 Client 并不在自己下联的 WLAN 下，而是通过 AP/Client 拥有的前缀发出请求，AP 同时还会向 AP/Client 发出提醒信息，AP/Client 收到提醒之后同时修改路由表。（2）IVIAnswer。在 AP 已经在后端处理完数据库的匹配和更新操作之后，AP 向 Client 回应请求。在请求中，AP 将可以使用的 IVI 地址进行附在 IVIAnswer 的 HTTP 数据包中。Client 在收到对应的数据包后，对自己的地址进行配置。（3）IVIAcknowledge。为了保证地址的正常工作，Client 在收到 IVIAnswer 之后，还会向 AP 反馈自己的地址配置结果。但 AP 确认对应的 IVI 地址已经成功的被配置之后，在对应 IVI 地址数据库中的状态修改为“已使用”。同时，作为物联网设备上报自己存活状态的一部分，Client 还会每隔一段时间向 AP 发送 IVIAcknowl

edge 的 HTTP 数据包。（4）IVI 地址超时问题。当 AP 长时间没有收到一个已经使用的地址时，AP 会主动的将对应的状态由“已使用”更改为“已分配”。

通过上面的设计中，系统成功的实现了 IVI 地址的分配。

5.2.2 地址取回机制

现在需要考虑的问题是：当设备重新连接的时候，如何才能取回原本的 IVI 地址。首先介绍一下 PID。PID 这个名字的来由是 Pi（树莓派）+ID。PID 在每个树莓派在第一次执行地址申请的时候自动生成，通过选用合适的随机生成方法（比如 MAC 地址加树莓派操作系统总运行时间）确保每一个树莓派的 ID 是独一无二的。在生成之后，这个 PID 将与树莓派终身绑定，除非人为使用 root 权限来进行修改。

下面考虑三种常见的重连情况。（1）断电重启。AP 在收到 IVIRequest 的时候发现了匹配的 MAC 地址和 PID，将之前的 IVI 地址重新分配给 Client。地址成功取回。（2）树莓派 SD 卡损坏（包括操作系统损坏，程序运行出错等软件错误）。这个时候在修复的过程中，管理人员直接更换 SD 卡进行重启。AP 收到 IVIRequest 请求，发现了 MAC 地址（无线网卡仍然没有变化）匹配，同样将这个地址分配给 Client，同时更新地址对应的 PID。（3）当无线网卡损坏的时候，情景是相反的。

此外，PID 还具有很好的功能扩展功能。通过给 PID 不同的前缀，AP 可以辨识出物联网设备的不同属性。比如，一个家具物联网中有冰箱和电视等物联网设备。可以在冰箱物联网 Client 设备生成 PID 的时候，添加一个属于冰箱的独特前缀，这样，冰箱就可以被分配到对应的冰箱地址。在实际地址分配中，全局 IPv6 地址的前缀分配也可以使用 IVI 地址分配一样的方法。这样可以进一步保证 IPv6 前缀的稳定。

第6章　部署情况和总结

6.1 部署情况

根据上面的设计思路一个演示无线物联网系统被搭建了起来。系统模拟实现了一个竖井 AP，一个 AP/Client 和多个物联网 Client 设备的物联网网络。在地址上，实验网络使用的 IPv6 公网网段有：2001:da8:b100:101d::/64 和 2001:da8:b100:10ff::/64。在 IVI 地址上，系统分配了 2001:da8:e300:79c2:a88f::/80 网段，该地址段有 16 个 IVI 地址，分别是 2001:da8:e300:79c2:a88f:4000::/96，2001:da8:e300:79c2:a88f:4001::/96，2001:da8:e300:79c2:a88f:4002::/96，2001:da8:e300:79c2:a88f:4003::/96 一直到 2001:da8:e300:79c2:a88f:400f::/96。

我们的部署拓扑图如下：

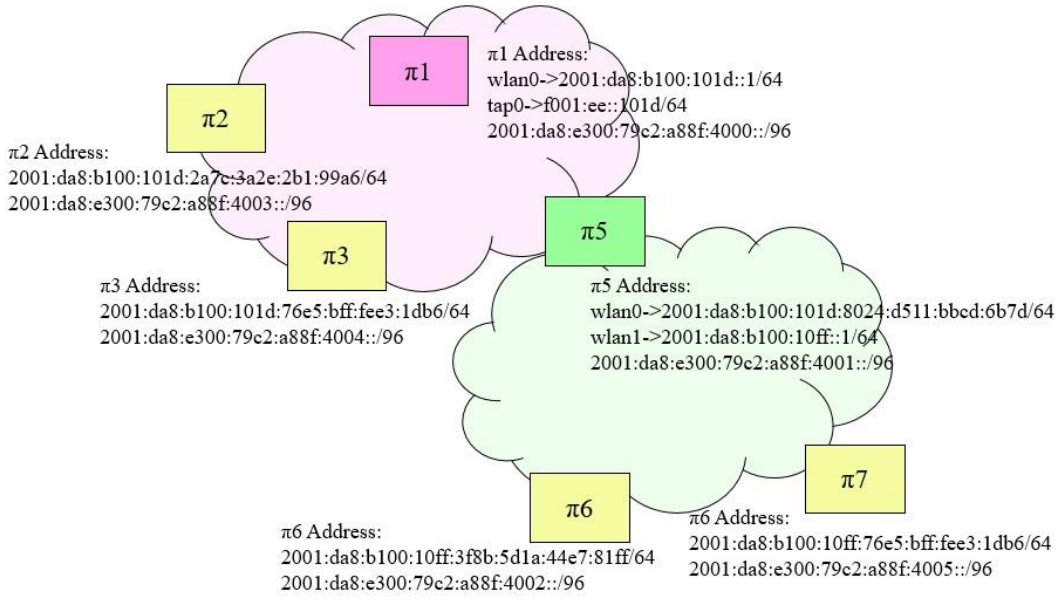

图6.1 部署结果拓扑示意图

在上面的部署中可以看到，在 π1 的虚拟 tap0 接口，所有的发往 IPv6 外网的数据包通过地址 f001::ee::101d/64 被 OpenVPN 封装放入隧道进行传输。

路由的部署模式图如下，在这里为了显示的简洁，选取一个典型的路由配置结果进行展示。在 AP 上路由表典型配置结果例子如下：(1)与直属子网路由配置：2001:da8:b100:101d::/64 dev wlan0。(2)与 AP/Client 下属子网路由配置：2001:da8:b100:10ff::/64 via 2001:da8:b100:101d:8024:d511:bbcd:6b7d/64 (3)与直属 Client 的 IVI 地址路由配置：2001:da8:e300:79c2:a88f:4003::/96 via 2001:da8:b100:101d:2a7c:3a2e:2b1:99a6 dev wlan0。(4)与 AP/Client 下属 Client 的 IVI 地址路由配置：2001:da8:e300:79c2:a88f:4002::/96 via 2001:da8:b100:101d:8024:d511:bbcd:6b7d dev wlan0。(5)与隧道服务器路由 default via f001:ee::1 dev tap0。通过下图可以得到一个部署后自动生成的路由的直观了解。

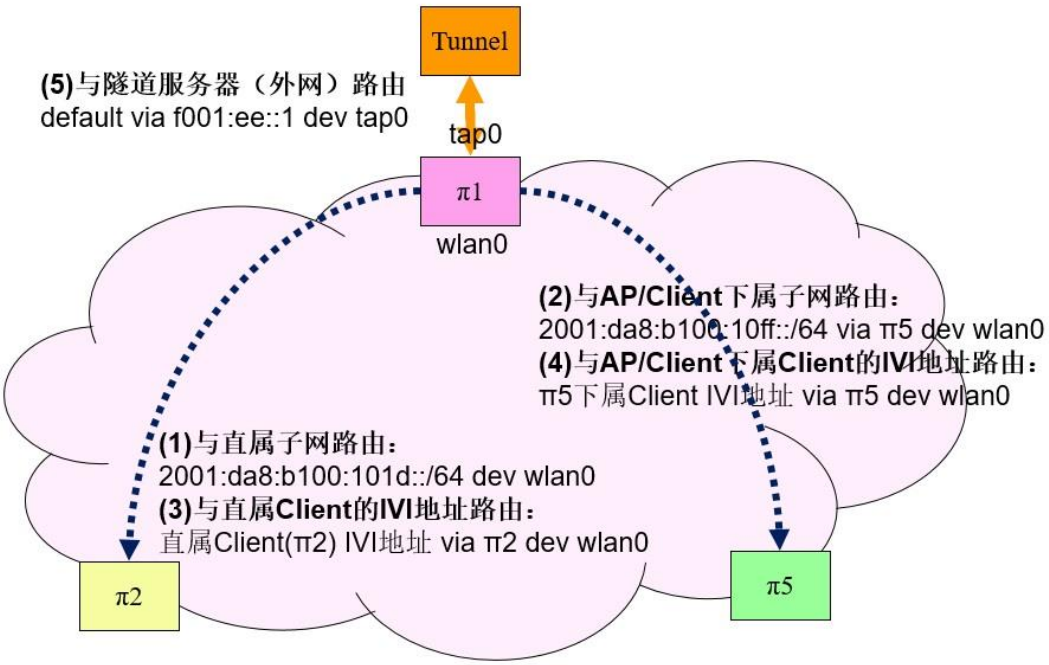

图6.2 AP部署路由结果图

Client 和 AP/Client 的路由配置结果相对简单。AP/Client 和 Client 想比，还需要负责进一步的子网路由和 IVI 地址路由。在隧道服务器上，路由自动配置与之对应，结果如下：（1）AP 子网路由：2001:da8:b100:101d::/64 via f001:ee::101d dev tap0。（2）AP/Client 子网路由：2001:da8:b100:10ff::/64 via f001:ee::101d d

ev tap0。（3）IVI 地址路由：2001:da8:e300:79c2:a88f::/80 via f001:ee::101d dev tap0。

在安全管理上，AP 和 AP/Client 使用了统一的基于 WPA2 加密的基于 IEEE802.11d 的 WiFi。具体的设置上，SSID 选择了不广播，这样对于非相关设备，SSID 是不可见的。最多可支持的用户数量是 20。在网页服务器上，选择了 SSL 加密。

截至 5 月 24 日，整个系统已经平稳工作了超过一周。系统实现了预期的物联网设备功能和网络结构。

6.2 网络性能测试

6.2.1 系统连通性和路由

在本章节中，无线物联网的内网外网连通性将得到展示。通过以下几个图片，我们将展示系统的路由过程。

```
pi@raspberrypi:~ $ traceroute 2001:250:3::191
traceroute to 2001:250:3::191 (2001:250:3::191), 30 hops max, 80 byte packets
 1  2001:da8:b100:101d::1 (2001:da8:b100:101d::1)  17.312 ms  17.192 ms  17.064 ms
 2  f001:ee::1 (f001:ee::1)  18.197 ms  18.178 ms  18.151 ms
 3  2001:250:3::191 (2001:250:3::191)  25.491 ms  25.679 ms  25.681 ms
pi@raspberrypi:~ $ traceroute 2404:6800:4005:80a::200e
traceroute to 2404:6800:4005:80a::200e (2404:6800:4005:80a::200e), 30 hops max, 80 byte packets
 1  2001:da8:b100:101d::1 (2001:da8:b100:101d::1)  9.761 ms  22.243 ms  22.231 ms
 2  f001:ee::1 (f001:ee::1)  22.199 ms  22.248 ms  22.543 ms
 3  * * *
 4  2001:250:1::1 (2001:250:1::1)  22.051 ms  22.085 ms  22.008 ms
 5  2001:da8:257::65 (2001:da8:257::65)  22.023 ms  21.914 ms  21.922 ms
 6  * * *
 7  2001:252:0:101::2 (2001:252:0:101::2)  52.377 ms * *
 8  2001:7fa:0:1::ca28:a10a (2001:7fa:0:1::ca28:a10a)  51.611 ms  51.504 ms  51.502 ms
 9  2001:4860:0:1::146d (2001:4860:0:1::146d)  52.804 ms  52.796 ms 2001:4860:0:1::1473 (2001:4860:0:1::1473)  52.779 ms
10  2001:4860:0:1::6fd (2001:4860:0:1::6fd)  52.782 ms 2001:4860:0:1::6ff (2001:4860:0:1::6ff)  53.854 ms  53.683 ms
11  2404:6800:4005:80a::200e (2404:6800:4005:80a::200e)  53.636 ms  53.533 ms  56.677 ms
```

图6.3 AP下属Client路由过程结果图

在上面的结果图中，处于 AP 下直接连接的物联网设备可以通过 AP 直接与外网相连。在第一项路由中，设备尝试连接本实验室搭建的 NMS。可以看到，Client 通过 AP 下 WLAN，再经由 AP 上搭建的隧道（f001:ee::1）到达 IPv6 外网。

在测试的第二部分中，我们向谷歌的 IPv6 镜像网站发起连通性测试。和之前测试同实验室的 NMS 不同，这里涉及到了更加复杂的路由活动。可以看到，访问谷歌时延时性比较明显。可见，AP 下属的 Client 的路由情况和设计的目标吻合。连通性和路由正确性得到了验证。

在下一张图中，AP/Client 下属子网的 Client 设备得到了验证。可以看到，除了和 AP 下属 Client 类似的路由路线，这里的 Client 还需要通过 AP/Client 的一次路由，即经由 2001:da8:b100:10ff::1 的 AP/Client 进行了一次路由转发。

```
pi@raspberrypi:~ $ traceroute 2001:250:3::191
traceroute to 2001:250:3::191 (2001:250:3::191), 30 hops max, 80 byte packets
 1  2001:da8:b100:10ff::1 (2001:da8:b100:10ff::1)  5.276 ms  4.789 ms  4.567 ms
 2  2001:da8:b100:101d::1 (2001:da8:b100:101d::1)  6.911 ms  6.689 ms *
 3  f001:ee::1 (f001:ee::1)  7.669 ms  21.654 ms  21.705 ms
 4  2001:250:3::191 (2001:250:3::191)  21.346 ms  21.031 ms  20.140 ms
pi@raspberrypi:~ $ traceroute 2404:6800:4005:80a::200e
traceroute to 2404:6800:4005:80a::200e (2404:6800:4005:80a::200e), 30 hops max, 80 byte packets
 1  2001:da8:b100:10ff::1 (2001:da8:b100:10ff::1)  7.880 ms  7.747 ms  7.367 ms
 2  2001:da8:b100:101d::1 (2001:da8:b100:101d::1)  29.525 ms * *
 3  f001:ee::1 (f001:ee::1)  28.863 ms  28.476 ms  28.140 ms
 4  * * *
 5  2001:250:1::1 (2001:250:1::1)  26.661 ms  26.527 ms  26.223 ms
 6  2001:da8:257::65 (2001:da8:257::65)  25.869 ms  31.356 ms  31.304 ms
 7  * * *
 8  2001:252:0:101::2 (2001:252:0:101::2)  79.984 ms * *
 9  2001:7fa:0:1::ca28:a10a (2001:7fa:0:1::ca28:a10a)  103.098 ms  103.192 ms  102.842 ms
10  2001:4860:0:1::146d (2001:4860:0:1::146d)  90.220 ms  89.759 ms 2001:4860:0:1::1473 (2001:4860:0:1::1473)  80.214 ms
11  2001:4860:0:1::6fd (2001:4860:0:1::6fd)  55.855 ms 2001:4860:0:1::6ff (2001:4860:0:1::6ff)  57.474 ms  56.063 ms
12  2404:6800:4005:80a::200e (2404:6800:4005:80a::200e)  56.571 ms  57.126 ms  64.424 ms
```

图6.4 AP/Client下属Client路由过程结果图

6.2.2 系统网络测试

在下面的内容中，通过测试系统不同设备的延迟特性和带宽，我们进一步的阐述系统的可行性。这里要特地指出来的是，物联网设备放置的位置不同，对延迟特性将会有很大的影响。因此，下面测量的数据更多的是对系统的表现进行定性的分析，说明系统设计的可行性和可靠性，而不是研究系统的一般表现。

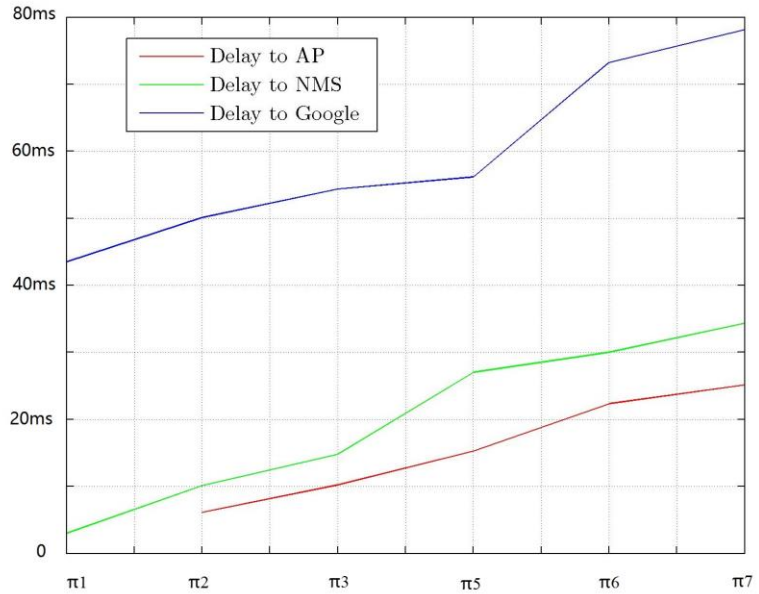

图6.5 设备网络延迟曲线

在上面的图中，物联网中不同的设备到 AP，NMS 和 google 镜像服务器的延迟得到了考察。在图表中，树莓派在横坐标上按照延迟时间快慢进行了排序，三条曲线分别是（1）蓝线：访问 Google 服务器。（2）绿线：访问 NMS 的延迟。（3）红线：访问 AP 的延迟。

从图中我们可以看到，系统中设备对外网的访问延迟在可允许的范围中。设备放置的相对位置不同，从红线中可以看出，延迟也会发生很大的变化。在我们设计距离内，延迟始终都是可以接受的。

此外，可以看到，整个无线物联网内部的网络延迟，想比 IPv6 外网连接带来的延迟也是相对较小的。

下图从带宽上对系统进行了分析。可以看到，带宽损失很大一部分来自于从 AP 到 NMS 上的隧道通路。无限物联网提供的带宽，大于 5Mbits/s，属于可以接受的使用带宽。在下一步的工作中，还存在进一步优化的可能。

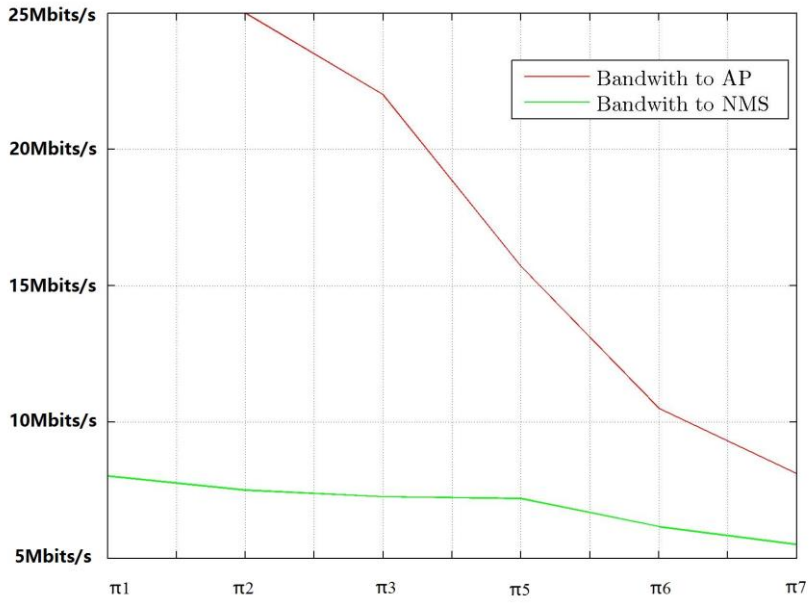

图6.6 设备网络带宽曲线

6.2.3 补充性数据

在前期的工作中，本次毕设还就 Ad Hoc 网络和 Infrastructure 网络的特性进行了验证。实验结果和前文提到的文献结果一致，Ad Hoc 网络的延迟要大于同等信道条件下的 Infrastructure 网络，测量的结果如图 6.6。在测量时，我们使用了

烧写了 OpenWrt 的 703N 路由器作为我们的实验平台（在功率控制上更加方便）。第一种 Ad Hoc 模式中，所有设备呈线性放置。而在第二种 Ad Hoc 模式中，所有物联网设备呈环形放置。所有的测试条件下，设备上的无线功率保持一致。除此之外，我们还注意到，Ad Hoc 模式下的设备在超过 5m 后的连接已经非常不稳定。同时 Ad Hoc 的建立连接时间也远远大于 Infrastructure 模式。

同理，我们也测量了基于 IPv6 和 IPv4 协议的物联网网络延迟和带宽，两者没有显著差别。这印证了我们网络二三层设计的合理性。

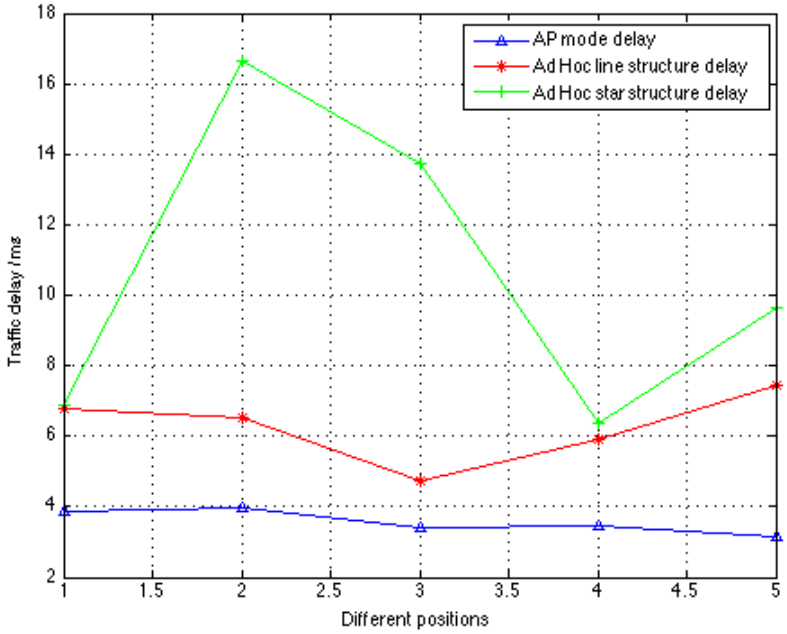

图6.7 Ad Hoc网络和Infrastructure网络延迟曲线图

6.3 总结和创新点

在本次毕设的物联无线网中，设计给出了一整套思路清晰，实际使用可靠简洁的物联网设计。本次毕设的创新点包括：（1）构建了一整套基于 IPv6 的无线物联网系统。毕设的无线物联网包括了从硬件平台，操作系统，硬件驱动，一直到应用层网页服务器的全套设计。物联网设备做到了自动连接物联网，自动配置地址和自动反馈设备运行状态。（2）通过使用 6over4 隧道，系统实现了在不支持 IPv6 的上游环境下配置 IPv6 物联网，通过隧道实现了 IPv6 物联网和外网 IPv6 的连接。（3）通过 IVI 地址翻译技术，我们实现了 IPv4 环境下用户对 IPv6 物

联网设备的访问。（4）给出了一套适应无线物联网设备的 IVI 地址分配机制。在这套分配机制中，我们实现了物联网设备断电重连和修复重连和地址分配的一致性。

6.4 下一步工作

下一步的工作可以从以下几个方向展开：（1）合法 MAC 地址数据库管理。虽然系统使用的 WPA2 加密协议和 SSL 协议能够保障通信的安全可靠，但是对地址进行管理可以进一步提高系统的安全性。（2）压力测试。随着连接管理的设备数量增加，平均通信能力也会下降。现阶段，因为树莓派设备数量的有限，部署的物联网设备还相对有限。在接下来的阶段，系统开发可以注意测试网络在设备数量很大的情况下的表现。（3）备用设备。在网络中，AP 或者 AP/Client 可能会因为电量或者损坏而下线，如果等待维修，在这段时间，数据传输功能将会发生失效。因此，如果系统中存在备用设备，就可以做到快速的切换，增加程序的可靠性。备用设备需要监控主设备，并且备份主设备的数据。在检测到主设备失效的时候，备用设备将快速的替代主设备功能。（4）协议修改。实际上，在物联网中，IPv6 的数据包头存在一定的冗余。IPv6 的数据头还可以根据物联网的特性进行进一步的修改和定制。物联网设备功耗较低，电源有限。通过这样的方法，系统可以有效的提高传输效率。

插图索引

图 2.1 几种常见的无线物联网结构图，图片来自[13]..... 5
图 3.1 无线物联网络设计拓扑图 10
图 3.2 IPv4 和 IPv6 发起的对话示意图 12
图 3.3 基于 OpenVPN 的隧道链接示意图..... 13
图 3.4 IVI 地址翻译示意图..... 15
图 4.1 AP 设备工作流程示意图 17
图 4.2 生成 SLAAC IPv6 地址的基本流程..... 18
图 4.3 Client 设备地址配置工作流程示意图 20
图 4.4 Client 设备数据搜集结果示意图 21
图 4.5 AP/Client 与 AP 子网前缀拓扑图 24
图 4.6 NMS 控制界面效果图..... 25
图 5.1 IVI 地址分配协议示意图..... 27
图 6.1 部署结果拓扑示意图 29
图 6.2 AP 部署路由结果图 30
图 6.3 AP 下属 Client 路由过程结果图 31
图 6.4 AP/Client 下属 Client 路由过程结果图 32
图 6.5 设备网络延迟曲线 32
图 6.6 设备网络带宽曲线 33
图 6.7 Ad Hoc 网络和 Infrastructure 网络延迟曲线图..... 34

參考文獻

[1] Atzori, Luigi, Antonio Iera, and Giacomo Morabito, “The Internet of Things: A survey”, Computer Networks, 2010.

[2] Ericsson, “Ericsson Mobility Report: On the Pulse of the Networked Society”, Ericsson Mobility Report, March 11, 2014

[3] Tsirtsis, G., and P. Srisuresh, “Network Address Translation - Protocol Translation (NAT-PT)”, RFC 2766, 2000.

[4] Barayuga, Vitruvius John D, and William Emmanuel S Yu, “Study of Packet Level UDP Performance of NAT44, NAT64 and IPv6 Using Iperf in the Context of IPv6 Migration”, International Conference on IT Convergence and Security (ICITCS), 2014.

[5] Timothy Rooney, “IP Address Allocation,” IP Address Management Principles and Practice, 2011.

[6] R. Roman, P. Najera and J. Lopez, “Securing the Internet of Things”, IEEE Computer, 2011.

[7] Li, Zhang, and Tong Xin, “Threat Modeling and Countermeasures Study for the Internet of Things.”, Journal of Convergence Information Technology, 2013.

[8] L. Coetzee and J. Eksteen, "The Internet of Things - promise for the future? An introduction," IST-Africa Conference Proceedings, 2011.

[9] Reina, D. G., Toral, S. L., Barrero, F., Bessis, N., and Asimakopoulou, E., “The Role of Ad Hoc Networks in the Internet of Things: A Case Scenario for Smart Environments”, Internet of Things and Inter-cooperative Computational Technologies for Collective Intelligence, 2013.

[10] D. Domingos, F. Martins, R. Martinho and M. Silva, “Ad-hoc changes in IoT-aware business processes,” Internet of Things (IOT), 2010.

[11] Li, J., Blake, C., De Couto, D. S., Lee, H. I., and Morris, R, “Capacity of Ad Hoc wireless networks”, ACM/IEEE International Conference on Mobile Computing and Networking, 2001.

[12] Chen, T., Wei, H., Hsu, N., and Shih, W., “A IoT Application of Safe Building in IPv6 Network Environment”, Computer Software and Applications Conference, 2013.

[13] Savolainen, Teemu, Jonne Soininen, and Bilhanan Silverajan, “IPv6 Addressing Strategies for IoT”, IEEE Sensors Journal, 2013.

[14] S. Deering and R. Hinden, “Internet Protocol, Version 6 (IPv6) Specification”, RFC 2460, 1998

[15] Z. Li, D. Zheng and Y. Ma, “Tree, Segment Table, and Route Bucket: A Multi-Stage Algorithm for IPv6 Routing Table Lookup”, 26th IEEE International Conference on Computer Communications (INFOCOM), 2007.

[16] M. Jung, C. Reinisch, and W. Kastner, “Integrating building automation systems and IPv6 in the internet of things”, Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2012.

[17] M. Eteläperä, M. Vecchio and R. Giaffreda, “Improving energy efficiency in IoT with re-configurable virtual objects”, IEEE World Forum on Internet of Things (WF-IoT), 2014.

[18] A. Dunkels, J. Eriksson, N. Finne, F. Osterlind, N. Tsiftes, J. Abeille, and M. Durvy, “Low-power IPv6 for the internet of things”, Ninth International Conference on Networked Sensing Systems (INSS), 2012.

[19] S. Ziegler, C. Crettaz, and I. Thomas, “IPv6 as a global addressing scheme and integrator for the internet of things and the cloud”, 28th International Conference on Advanced Information Networking and Applications Workshops (WAINA), 2014.

[20] A. J. Jara, D. Fernandez, P. Lopez, and M. A. Zamora, “Lightweight mobile IPv6: A mobility protocol for enabling transparent IPv6 mobility in the internet of things”, Global Communications Conference (GLOBECOM), 2013.

[21] L. Mainetti, L. Patrono, and A. Vilei, “Evolution of wireless sensor networks towards the internet of things: A survey”, Software, Telecommunications and Computer Networks (SoftCOM), 2011.

[22] Chris Heegard, “Range versus Rate in IEEE 802.11g Wireless Local Area Networ”, Draft, 2001.

[23] Feilner, Markus, “OpenVPN: Building and Integrating Virtual Private Networks: Learn how to build secure VPNs using this powerful Open Source application”, 2006.

[24] Shuai Yang, Qianli Zhang and Xing Li, “A Tunnel Broker based on IPv6 access system for a small scale network with IPv4 Upstream”, ITnec, 2016.

[25] Li, X., Bao, C., Chen, M., Zhang, H., and Wu, J., “The China Education and Research Network (CERNET) IVI Translation Design and Deployment for the IPv4/IPv6 Coexistence and Transition”, RFC 6219, 2011.

[26] Shang, Wentao, Congxiao Bao, and Xing Li, “IVI-based Locator/ID Separation Architecture for IPv4/IPv6 Transition”, International Conference on Networking, 2012.

[27] Tews, Erik, and Martin Beck, “Practical attacks against WEP and WPA”, Wireless Network Security, 2009.

[28] Thomson S, Narten T. “IPv6 Stateless Address Autoconfiguration”, RFC 4862, 1996.

致谢

感谢李星教授、包从笑老师对我的耐心指导！在毕设期间，我作为一个对网络一窍不通的初学者，学习到许多宝贵的知识！

感谢杨帅师兄提供的隧道源代码和隧道代理服务器！杨帅师兄稳定而可靠的工作，是我毕设的基础。同时，杨帅师兄提供了非常非常多的网络方面的调试帮助和经验。感谢盛茂家师兄，同样提供了许多帮助，盛茂家师兄的 IVI 服务器，性能非常出众。两位师兄，无论是做人做事，都是我学习的榜样！

感谢和我一起毕设的王文鑫同学。在毕设期间，我们互相鼓励，互相讨论学习。没有他的帮助，我很难坚持完成这一个巨大的项目。

感谢清华大学的培养！

声　明

本人郑重声明：所呈交的学位论文，是本人在导师指导下，独立进行研究工作所取得的成果。尽我所知，除文中已经注明引用的内容外，本学位论文的研究成果不包含任何他人享有著作权的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确方式标明。

签　名：王彦行　　日　期：2016.6月4日

附录 A 外文资料的调研阅读报告或书面翻译

文献翻译：Evolution of Wireless Sensor Networks towards the Internet of Things: a Survey 无线传感网络向物联网络的演变：一项调研

摘要

无线传感器网络（WSN）正在包括医疗保健，农业，环境监控和智能电表的多个应用场景发挥越来越关键的作用。不仅如此，无线传感器网络具有很强的网络异构性，因为它由许多不同架构的私有网络和公共网络的组成。这样的多样性使得在现有的传感网络上部署和集成新技术变得十分复杂。目前的学术趋势是摆脱专有和封闭的网络标准，使用新的以 6LoWPAN/IPv6 为基础的无线传感器网络。这项技术将使无线传感器网络与互联网之间的本地连接成为可能，从而使智能对象参与到物联网之中。从头开始建立一个全 IP 的物联网是很困难的，因为许多不同的传感器和执行器技术（有线和无线）已经被部署和使用了许多年。在对现有的最为先进的技术进行回顾之后，本文提出了一个能够协调传统架构和全 IP 环境系统的可能方案。我们将在本文中使用楼宇自动化的情景来讨论和测试这个框架的潜在好处。

1. 导论

未来的互联网旨在整合包括有线和无线的异构通信技术，从而实现物联网。虽然有许多方法来描述物联网，但我们可以将其定义为一个基于标准通信协议的，可唯一寻址互连对象的全球网络。传感器技术的低成本已经促进了许多应用性场景的发展，比如环境监测，农业，医疗和智能建筑无线传感器网络。无线传感器网络的特点是高异构性，因为它们同时拥有不同的私有和非私有的子系统。网络 http://info.tsinghua.edu.cn 广泛的异构性，现如今阻碍了我们大规模部署一个整合了所有现有的传感器网络的虚拟广域传感器网络的工作。因此异构传感的互操作性，和低层（即硬件）和高层次（即用户应用程序）之间的抽象层次化是非常重要的课题。基于封闭和专有系统的传感器网络就像一些分散的群岛，他们和外部世界进行的沟通非常有限。通常情况下，我们需要使用知道应用程序特定信息的网关，才能将数据导出到连接到因特网的其它设备。此外，

除非使用网关或代理服务器来处理复杂应用中特定的数据转换，不同无线标准的系统无法直接通信。现在的趋势是使用因特网协议来实现无线传感网络和因特网的本地连接。这样的话，智能对象（比如拥有网络接口的小型传感器和执行器）就能够被互联从而实现物联网。这主要需要我们使用开放式协议，同时也需要所有的设备都有它自己的 IP 地址。物联网可以让我们收集智能对象在它生命周期中从客观物理世界中的观测到的各种各样的信息。让智能对象连网，我们就可以让未来的因特网拥有更加大的灵活性，定制更加多的用户设计。比如，在网络混搭设计的潮流中，用户可以编写应用程序来把家电这样的真实设备和网络上的虚拟设备关联起来进行管理。这种应用程序我们通常叫做物理混搭设计。本文的工作提供了关于构建物联网现阶段的标准和无线传感器网络的解决方案的综述。此外，本文提出了一个能够协调传统架构和全 IP 环境系统的可能解决方案。我们使用楼宇自动化的情景来讨论和测试这个框架的潜在好处。全文的剩余部分是按照下面的方式组织的。在第二章中，我们将描述现有物联网的情景和遇到的问题。在第三章中，我们将对现有的无线传感器和执行器网络使用的技术进行一个综述和分析。最后，本文将在第四章对楼宇自动化的例子分析和测试。这将帮助 我们理解我们构建网络时的各项需求。第六章，我们将总结全文。

2. 情景和挑战

物联网在一下的情景中都有非常重要的作用：1)医护和医疗设备。2)家具和楼宇自动化。3)提高能量效率，节能。4)工业自动化。5)智能电表和智能电网。6)环境监控和天气预报。7)更加灵活的射频识别技术。8)资产管理和物流。9)运输和传输自动化。10)农业。11)智能购物。

广泛的应用，在应用要求、规模和总可用市场方面的巨大差异，导致了嵌入式网络领域技术解决方案的层出不穷。智能对象具有不同的通信，信息和处理能力，为了实现智能对象之间的无缝互动，互操作性非常重要。可扩展性是物联网的另外一个重要话题，因为沟通需要无缝互连来自一个从物体到人这样大的范围中的个体。此外，在网络无所不在的动态环境中，对象发出的服务必须自动的由发现机制来识别。保密性，真实性和通信的可信任性是个人隐私和安全性的保证（例如，在计费取决于传感器的数据大小的情况）。虽然一个单一的传感器或致动器需要传输的数据量是非常有限的，但是总量可以是巨大的。因为存在实际情况中，往往存在大量的对象和大量相互作用。因此如何处理大容

量 数据是未来互联网的重要挑战之一。此外，容错是物联网的另外一个问题。为 了保证可靠的通信，无处不在的网络需要冗余几个层次，并且需要能够自动适应环境变化。最后一点，当智能对象是电池供电的时候，我们需要考虑功耗。节能高效的通信机制是必不可少的。

3. 现有方法回顾

本章将会提供无线传感网络主要的技术的回顾。我们首先从不使用因特网协议的技术开始。然后我们会讨论 IPv6/6LoWPAN。最后我们会讨论高层技术和中间件。

3.1. 非 IP 方法

1) ZIGBEE： ZigBee，又称紫蜂协议，是由 ZigBee 联盟开发的一种低数据速率和短距离应用的无线联网技术。ZigBee 协议栈是由四个主要层：物理（PHY）层， 媒体访问控制（MAC）层，网络层（NWK）和应用程序（APL）层组成的。ZigBee 协议中的 PHY 和 MAC 层根据 IEEE 802.15.4 标准定义，而其余的协议栈由 ZigBee 自己规范定义。在 ZigBee 基于的 IEEE 802.15.4 的初始版本中，工作频率为 868MHz（欧洲），915MHz（北美）和 2.4GHz（全球）。数据传输速率分别是 20kb/s，40kb/s，和 250kb/s。 特别值得注意的是，ZigBee 网络层支持寻 址和树、网状拓扑路由。ZigBee 应用的开发依赖于调整应用程序的设置。最重要的 ZigBee 应用规范是 ZigBee 家庭自动化公共应用规范和 ZigBee 智能能源规范。家庭自动化的主要应用领域是照明，空调和安全。智能能源规范则涉及应用电网中的能源需求响应和负载管理。新的 ZigBee 规范是 RF4CE。它的网络协议栈被简化，只支持星型拓扑结构。它可以给消费类电子产品遥控一个简单的解决方案。

2) Z-WAVE： Z-Wave 是指 Zensys 公司推动的在住宅和小型商业环境运用场景下的 ZWave 联盟自动化无线协议架构。Z-Wave 的主要目的是支持从一个控制单元向一个或多个节点的短消息可靠传输。Z-Wave 根据五个主要的层构成的结构进行组织：物理层，媒体访问控制层，传输层，路由层和应用层。Z-Wave 主要工作在 900MHz（在欧洲 868MHz，在美国 908MHz）和 2.4GHz。ZWave 额定传输速率为 9.6kb/s，40kb/s 和 200kb/s。Z-Wave 定义了两种类型的设备：控制器和受控器。控制器发送命令给受控器，而受控器负责执行命令。Z-Wave 的路由层使用基于源的算法进行路由。

3) INSTEON：INSTEON 由 INSTEON 联盟开发用于家庭自动化，随后由 SmartLabs 加以促进，发扬光大。INSTEON 其中一个显着特点是，它定义了一个由射频器件和电源线连接组成的网状拓扑结构。设备可以只含射频器件或者电源线，也可以支持两种混合的通信。INSTEON 射频的工作的中心频率为 904MHz，额定数据传输速率为 38.4kb/s。INSTEON 设备是相互平等的，这意味着它们中的任何一个都可以成为发送器，接收器，或延迟器。不在同一个通信范围内的设备间的通信通过一个依赖时隙同步的多跳方式来实现。

4) WAVENIS：Wavenis 是由 Coronis 系统公司开发的一个无线协议栈。它主要用来控制和监控包括家庭和楼宇自动化在内的无线运用情景。Wavenis 目前由 Wavenis 开放标准联盟（Wavenis-OSA）管理和推广。在协议中定义了物理层，链路层和网络层的功能。Wavenis 支持上层通过应用编程接口（API）来进行访问。Wavenis 主要工作在 433MHz，868MHz 和 915MHz 频段。这几个频段是亚洲，欧洲和美国的 ISM 频段。有些产品还运行在 2.4GHz 频段。Wavenis 提供的最小和最大数据传输速率为 4.8kb/s 和 100kb/s。19.2kb/s 的是典型的数据传输速率。

3.2. 基于 IP 的方法

尽管互联网架构传感器网络的适用性遭到了许多研究者的最初的怀疑，但是今天总的趋势是从专有或封闭的标准解决方案走向使用 IP 技术的解决方案。事实上，最近 32 位微控制器性能的进步和高度优化的协议堆栈，使得将 IP 连接添加到智能对象不再是不可能完成的任务了。ZigBee 联盟在智能能源 2.0 配置协议中使用了 IP，也进一步证实了这一趋势。考虑到我们将要连接的潜在设备数量可能是非常巨大的（爱立信公司估计将大于 500 亿个），IPv4 因为其有限的地址空间是不能被我们采用的。更好的选择当然是使用 IPv6，因为它拥有 128 位地址。除此之外，IPv6 还允许网络自动配置和无状态操作。由于 IPv6 中，每一个智能对象可以很容易的双向连接到其他基于 IP 的网络。在这个过程中，我们不需要翻译网关或代理这样的中间实体。鉴于存在数据包大小限制和低功耗无线个人区域网络中其他的限制，我们使用一个适配层进行对应的 IPv6 报头压缩，分段和地址自动配置。IETF 的 6LoWPAN 工作组已经确定了 IPv6 和 IEEE 802.15.4 之间适配的格式。理想使用 6LoWPAN 的运用场景是这样的：应用嵌入式通过它在保证移动通信可靠性的同时，访问整个使用开放协议的大尺度的网络结构。在图 2 中是 6LoWPAN 的协议栈。6LoWPAN 由低功率无线区域网络（WPA

N）组成，它们通过边缘路由器连接到其它 IP 网络。边缘路由器通过处理 6LoWPAN 的压缩和邻居发现来完成进出 LoWPAN 流量的路由。如果 LoWPAN 连接到 IPv4 网络，边缘路由器也负责处理对 IPv4 的互连。每个 LoWPAN 节点都有唯一的 IPv6 地址来供其他节点识别，并且能够发送和接收 IPv6 数据包。通常情况下 LoWPAN 节点支持 ICMPv6 的流量。它可以使用 ping 功能，并且使用用户数据报协议（UDP）作为传输协议。通常，IPv6 和 LoWPAN 格式之间适配和转换是通过在 6LoWPAN 的网络边缘的路由器完成的。这些路由器简称为边缘路由器。这种转变在两个方向上看，都是是透明，高效，无状态的。此外，6LoWPAN 的不依赖特定的基础结构来完成操作，但也可以工作在 ad hoc 模式下。在写这篇文章的时候，IETF 路由选择低功耗和有损网络（ROLL）工作组正在定义低功耗和有损网络（RPL）的 IPv6 路由协议。

3.3 高层次和中间体解决方法

毫无疑问，一个基于 IP 协议互连相关的网络的主要优势是能够使用 Web 服务。近日，IETF 的另一个工作组已经提出了一个基于智能对象的网络的 Web 服务。此协议称为约束应用协议（COAP）。它用来支持在能源资源有限，高分组丢失率，硬件能力有限的特殊的环境中运行 Web 服务。它尤其是用来支持物联网中的对应服务。COAP 为了物联网这样资源受限的网络及机器对机器（M2M）应用而做了优化。COAP 是超文本传输协议（HTTP）的功能子集，它被重新设计以便可以在小型嵌入式设备使用。这样的只拥有低处理能力和并且有能量消耗约束的小型设备包括传感器和致动器。COAP 被组织在两个子层（即请求/响应层和事务层），建立在用户数据报协议（UDP）的顶部。COAP 使用 4 字节跟随紧凑二进制选项的短固定长度二进制标头。COAP 支持几种有效载荷编码标准，例如可扩展标记语言（XML）。图 2 显示了 COAP 的一个完整的协议栈。虽然 COAP 还在进行并没有完成，但是几个开源实现已经公布了，其中包括两个著名的无线传感器网络操作系统：Contiki 和 TinyOS。

采用全 IP 架构对 COAP 来说是最合适的。当需要传统技术解决方案和基于 IP 的解决方案共存的情况下，一个可行的选择是采用高层次的中间件，从而提高灵活性和互操作性。这样就可以支持各种应用程序。在理想的情况下，这样的中间件应能兼容所有从互联网骨干网继承的 Internet 老式架构。

全球传感器网络（GSN）是一个满足上述要求，用 Java 编程语言开发的一个开源框架。GSN 的一个关键概念就是虚拟传感器抽象。虚拟传感器对应一个数

据流。这个数据流要么是直接从传感器接收的数据，要么是其他虚拟传感器导出的数据流。在后一种情况下，使用对等覆盖网络可以提高通信性能。虚拟传感器组成是：

- 一个封装：包含从一种数据源读取数据的逻辑功能的类。

- 处理类：拥有后处理逻辑功能的一个或多个类。这意味着，当一个封装从数据流中读到数据后，后处理的代码将会被执行来处理它们。

- 一个描述符文件：一个用于配置虚拟传感器的 XML 文件。在这个文件中，一般包含有关虚拟传感器的信息，用于描述特定的虚拟传感器的地理定位数据的描述信息，比如纬度经度。这个文件也包含其他种类的信息。

GSN 使用基于 SQL 的查询来创建数据流，这样就可以把数据从传感器网络资源传送到利用互联网作为骨干网的数据用户手上。该架 4 构必然是分布式的。此外，GSN 还配有传感器发现协议，它包括查询，订阅，和登记。通过数据聚合在中间节点组合、传递信息，可以减少网络流量负荷。GSN 能够聚集查询响应，探索新的传感器节点，并注册可用的传感器节点。GSN 采用面向服务的体系结构，并且不排除使用其他标准为的物联网，比如先前所描述 6LoWPAN 架构。

4. 楼宇自动化的案例测试研究

为了说明上述的技术的潜在效益，我们选定一个特定的参考情景进行案例研究。特别要注意的是，这项工作涉及楼宇自动化。物联网可以大幅度提高楼宇中传统能源管理系统的效率。我们已经知道，在欧洲能源消耗的 40％以上是与建筑相关的（住宅，公共建筑，商业建筑和工业建筑）。以先进，灵活，集成的信息通信技术为基础的能源管理系统，在不同的建筑中，都可以更好的控制自然采光和通风，保持更好的温度（通过控制窗户，地板和天花板）。这将减少能源消耗，同时也提高了用户和财产安全，提供了更加多的生活福利和辅助。

我们使用大学校园来展示不同种类的传感器和执行器网络融合的潜在价值。智能建筑能够节省能源，保持舒适度和改善工作环境。这样的大学校园的设计和控制是一个非常有趣的挑战。楼宇自动化旨在通过控制取暖，照明，遮阳，门/窗等许多异构设备，为用户提供真正的舒适，安全的环境，对多个环境进行监控。所有这些设备都可以看成一个宽 WSN 结构中的传感器和智能致动器。

大学建筑中不同的异构环境需要能源，如在教室，实验室，教授办公室，行政办公室和公用部分中。智能建筑应能尽量减少每一种能源的浪费。目前有各

种各样的，由不同的供应商提供的低成本的传感器和致动器，它们可以能够测量温度，湿度，空气质量，亮度和光度。不幸的是，这些传感器和智能表系统往往无法实现互操作。因此，我们需要一个基本的整体控制策略来连接所有的能源消耗部件（如采暖，空调，照明，能源生产和存储中的能耗部件等）。我们需要互操作性，可扩展性和足够的语义才能保证实现数据的独立性。只有这样，我们才能实施控制策略。

一个可能的满足上述要求的策略是采用一种在网络层基于 6LoWPAN/IPv6 的全 IP 解决方案，并在 COAP 上来实现对传感器和执行器的 Web 访问。然而如前所述，部署多年的异构的老架构（包括有线和无线）无法支持重新部署全 IP。考虑到这一点，我们提出了一个试图协调传统和新架构，可以在以后阶段再向全 IP 环境迁移的框架。具体而言，我们建议采用 GSN 作为 6LoWPAN/IPv6 网络和传统的无线传感器网络的统一中间件。中图 3 描绘了所提出的架构。

在此框架下，6LoWPAN 的边缘路由器装置具有双重功能：它连接的 6LoWPAN 到因特网，并且充当我们上述的 GSN 中间件的虚拟传感器。对于传统的非 IP 技术（例如 ZigBee 的 Z-Wave 等），每个 WSN 网关都需要实现一个特定的 GSN 虚拟传感器。传感器的数据和用户应用数据之间的依赖关系（比如在能量管理系统，人员跟踪系统，环境监测系统等系统中）是一个主要问题。如果采用的 GSN-基础的框架，事情就会变得更简单，因为它允许、保证了数据独立性。因为 GSN 的存在，用户应用程序可以从异构物理基础设备的细节中利用虚拟传感器的概念进行抽象。这些异构物理基础设备包括有线或无线传感器，RFID 读写器，智能表，摄像机，服务器。GSN 提供了传感器网络的逻辑覆盖，从而可以利用虚拟传感器概念抽象出信息。GSN 架构的一些模块能够进行如存储和储存库历史数据查询等基本任务。不幸的是，GSN 框架的当前版本有一些缺点。比如，每一次用户的应用程序的需求更改，我们都需要设计一个新的封装。为了减少这些问题，我们需要构建一个语义查询机制。语义 Web 服务是一个可以用于数据检索和动态发现服务的创新而且灵活的思路。当前 GSN 的另一个弱点是它的集中式架构，因此只支持点对点通信。我们可以这样修改 GSN：开发有效的，采用的对等方式（P2P）的发现服务，这样就能够提高虚拟传感器覆盖网络的管理。举个例子，我们可以使用分布式哈希表（DHT）。

5. 测试环境设置

在写这篇文章的时候，我们正在建立一个用来进行测试的环境。如前所述，在楼宇自动化的测试情景下，我们结合使用了 6LoWPAN/IPv6 的（网络层）和 GSN（作为一种更高层次的应用中间件）。测试的无线节点是图 4 中，基于意法半导体的 MB851 开发电路板。后者采用了 STM32W108 系统级芯片（SoC），集成了 32 位 ARMCortexTM-M3 的微处理器,24MHz，128KB 的闪存，8KB 的 RAM 和一个 2.4GHz 的 802.15.4 标准的收发器。在这个测试环境中，应用板运行 Contiki 程序。Contiki 是一个开源的内存高效的网络化嵌入式系统和无线传感器网络操作系统。Contiki 是专为只有少量的内存的微控制器设计的。典型 5 的 Contiki 配置是 2kb 的 RAM 和 40kb 的 ROM。通过嵌入式 uIPv6 子系统，它可以提供包括 IPv4 和 IPv6 的 IP 通信。uIPv6 实现了 IPv6/6LoWPAN 的堆栈，可以传输使用 STM32W108 芯片的 IEEE802.15.4 的无线电 IPv6 数据包。

试验中的网关设备是基于意法半导体的 SPEAr1310 芯片。它是一个多核多线程，可以运行 Linux 的最先进的嵌入式系统平台。网关将实现 6LoWPAN 的边缘路由器功能。这样我们就可以将无线传感器网络连接到因特网。这个网关还会通过 GSN 来管理和监控远程节点配置，引导，节点监控，链路状态，并且给出实时分析。在写这篇文章的时候，这个测试环境的实施才刚刚起步，所以还没有足够的细节信息。VII.总结现有的无线传感器网络技术部署多年来许多不同的专有和非专有解决方案，具有高度的异构性。这成为了学术界在对未来互联网传感器的进行普遍整合时，必须面对的一大挑战。目前的趋势是摆脱专有和封闭的协议标准，使用以 6LoWPAN/IPv6 的 IP 为基础的无线传感器网络。本文对现有的异构无线传感器网络的融合标准和解决方案进行了讨论。此外，本文作者提出了一个能够协调新架构和传统框架（即非基于 IP 的框架）。在这个框架中，我们可以保留在中后期再进行全 IP 环境迁移的可能性。这个框架目前正在楼宇自动化情景中进行测试。关于该框架的更多信息将在经过多次现场试验后的阶段予以展示。（原文 24348 个英文字符，4341 个英文单词）

原文索引：

[1] L. Mainetti, L. Patrono, and A. Vilei, “Evolution of wireless sensor networks towards the internet of things: A survey”, Software, Telecommunications and Computer Networks (SoftCOM), 2011.

综合论文训练记录表

<table>
<tr><td>学生姓名</td><td>王亭午</td><td>学号</td><td>2012011018</td><td>班级</td><td>无 210</td></tr>
<tr><td>论文题目</td><td colspan="5">一种无线物联网的设计与实现</td></tr>
<tr><td>主要内容以及进度安排</td><td colspan="5">一．开题答辩前时间安排

完成 OpenWrt 系统在路由器上编译和搭建。实现 IPv4 上的数个路由器 Open Wrt 物联网的 Ad hoc 网络搭建。

二．完成基于 IPv6 的物联网硬件搭建

在寒假期间，继续熟悉 OpenWrt 的架构，同时熟悉交叉编译的基本原理和操作。熟悉 IPv6 的环境搭建，准备好相应的硬件基础。在回到学校后，完成基于 IPv6 的物联网搭建。

三．路由算法的研究和分析

在实现 IPv6 下的物联网平台上，尝试测试不同的路由算法，记录对应的路由效率和鲁棒性。同时，根据 IPv6 的固有特性，设计低能耗，高鲁棒性的路由算法。在路由设计的时候，考虑到一些具有代表性的无线路由情景，能够设计出合适的基于 Ad Hoc 网络的 IPv6 路由算法。所设计的算法应该能够适应智能电表上的物联网运用。在设计出对应算法后，对对应的算法进行软件和硬件上相应的仿真和验证。

指导教师签字：
考核组组长签字：
2016 年 1 月 15 日</td></tr>
<tr><td>中期考核意见</td><td colspan="5">该同学的工作基本进入正规，已完成大量实验测试的工作，但尚需要加强开发工作，加快速度，以便正常完成预期目标。

考核组组长签字：
2016 年 5 月 30 日</td></tr>
</table>

指导教师评语	在调研和实验的基础上，设计并实现了一种与 IPv4 互通的 IPv6 无线物联网的方案和系统。达到了本毕设的目标。 指导教师签字： 2016年 5月30日
评阅教师评语	该论文设计并实现了在 IPv4/IPv6 共存的网络环境下，家庭物联网使用 IPv6 协议联网的地址分配和路由算法机制，并进行实验验证。取得了良好的实验结果。该论文具有重要的应用价值。论文结构清晰，思路明确，图表完整。达到了综合论文设计的水平 评阅教师签字： 2016 年 6 月 2 日
答辩小组评语	答辩讲述清楚，回答问题正确，论文结构合理，图表清晰。总体成果显著满足了综合论文的目标和要求 答辩小组组长签字： 2016年 6 月 15 日

总成绩：90

教学负责人签字：

2016 年 6 月 16 日