LoRa 技术开发指南

云服务篇

唯传科技
WINEXT
TECHNOLOGY

IOT 低功耗广域网与服务平台方案商
LPWAN OPERATOR PLATFORM FOR IOT
www.winext.cn

前言

1.1. 技术背景

我国进入 21 世纪后，城镇化建设的步伐不断加快，每年有上千万的农村人口进入城市。据世界银行城市可持续发展报告测算，以 100 万人口的中等城市为例，如开始智慧城市建设，当其达到实际应用程度的 75%时，该城市的 GDP 在投入不变的条件下产能增加 3.5 倍。这意味着智慧城市可促进经济增长翻两番，完全有可能实现“四倍跃进”的城市可持续发展目标。

智慧城市就是运用信息和通信技术手段感测、分析、整合城市运行核心系统的各项关键信息，从而对民生、环保、公共安全、城市服务、工商业活动各种需求做出智能响应。城镇化的加速发展，部分地区“城市病”问题日益严峻。为解决城市发展难题，实现城市可持续发展，建设智慧城市已成为当今世界城市发展不可逆转的历史潮流。

随着智慧城市的全面部署以及城市智能化、感知与互联的发展需求，城市越来越多的碎片化终端设备需要低功耗长距离传输的接入网络。业界预测到2020年物联网500亿个无线节点中只有不到10%的使用GSM技术，以LoRa为代表的低功耗、远距离网络技术的出现，有机会打破物联网在互联方面的瓶颈，促进物联网端对端的成本大幅下降，引爆物联网的大规模应用。

LoRa是由升特公司发布的一种专用于无线电调制调解的技术，相对一个开放的系统，具有低功耗、易组网、成本低、传输距离远等优势，可以满足长时间的运作，电池供电使用时间长达数年，故此技术在世界多地已部署。

截至目前最新公布的数据，已经有17个国家公开宣布建网计划，120多个城市地区有正在运行的LoRa网络，如美国、法国、德国、澳大利亚、印度等等国家，荷兰、瑞士、韩国等更是部署或计划部署覆盖全国的LoRa网络。

Orange、KPN、SK、TATA、软银、Senet、Comcast等各国主流电信运营商已选择LoRa来建设物联网专用网络，形成源于LoRaWAN的物联网标准规范并大范围推广。

美国网络运营商Senet已经在美国110个城市超过125000平方公里的面积搭建了LoRa试验网络，同时计划2017年在另外10个大型城市部署相关网络和服务，这将使得LoRa在美国覆盖23个州的225个城市，覆盖人口达到5千万。

由英国政府支持的Digital Catapult组织将在伦敦建设LoRa网络，这个网络将免费提供给中小企业使用；Digital Catapult将和BT（英国电信）合作在伦敦一些区域去建设50个LoRa基站。

韩国SK通信与540个合作公司一起部署的LoRa应用服务大概有50项，其中现在主推的是与韩国农村社区公司合作的农业相关水位测量的项目，还有包括单车追踪系统和定位监视方案，贸易区分析方案，智能路灯控制方案等。

唯传科技是一家专注与物联网低功耗广域网（LPWAN）与运营服务平台的方案商，自主研发了 AirNode 专利技术，为企业客户提供运营级低功耗的网络建设、设备连接、数据传输与平台服务。为实现物联网真正互联互通的目标，我们能提供全面技术方案与支持，帮助企业客户建立全新的商业模式，实现合作共赢。

唯传科技团队有多年的物联网研发经验，能够洞察并理解传统企业客户希望实现新技术改造的迫切需求。考虑到目前在传统行业进行互联网转型过程中普遍存在的各种现实困难，我们为客户打造了一揽子的解决方案。特别是开创性地提出了为客户定制自己的物联网私有云服务的解决方案，有效解决了各种应用场景存在的技术难题，为实现传统产业规模化技术改造、升级提供了非常好的解决思路。本文档正是基于以上考虑，从基本概念和实践应用，从解决实际问题的角度，系统地从唯传科技的云服务所牵涉的技术路线、解决方案、实施步骤等方面做了深入浅出地描述。

唯传科技充分理解用户在开发物联网应用中遇到的各种实际困难，推出了《LoRa 技术应用开发指南》系列文档，涵盖了设备开发、云服务、测试、工程安装调试等领域，是迄今为止国内业界最全面的 LoRa 技术指南。本文档是其中的一个子篇目，专注在云服务器的操作和使用说明上，帮助用户可以快速地上手唯传科技提供的私有云中间件服务。由于唯传科技封装了硬件层面的实现细节，并提供了丰富的调试诊断工具，因此可以极大提升了用户使用和开发自己的物联网系统的效率，为推广低功耗广域网 LPWAN 技术作出了有益的贡献。

目前，唯传科技已研发出基于 AirNode 技术的 AN-M100A 传感器前端接入节点、AN-GW5000 网关路由器、云中间件服务等产品。同时唯传科技也作为 LoRa 国际联盟会员，将致力于推动 LPWAN 的标准化进程和产业的健康发展。

1.2. 致谢

本文档是唯传科技研发团队全体同仁的集体智慧结晶，特别需要感谢以下贡献人员：

Tony Zhang
Ss.Guo
W.Wang
Jx.Cao
Zn Chen
Hl.zhou
Jessie Hou
RainStar.He
Zz.Wu

1.3. 版权申明

深圳市唯传科技有限公司

2017.5

名词解析

本章重点对一些术语进行阐述，便于用户能够对一些技术名词有一些基本的了解。

LoRa

LoRa 是低功耗广域网通信技术中的一种，是 Semtech 公司采用和推广的一种基于扩频技术的超远距离无线传输技术，是 Semtech 射频部分产生的一种独特的调制格式。

LoRaWAN 协议

LoRaWAN 是基于 LoRa 的低功耗广域网，它主要包括 2 个部分：**通信协议和体系结构**。它能提供一个：低功耗、可扩展、高服务质量、安全的长距离无线网络。

设备激活

为了实现前端节点设备或网关和后台的正常数据通信，需要注册一个节点或网关到后台服务器，此时需要对前端设备设置相关的ID标识字串，激活就是使用这些字串与后台云服务器进行身份校验的一个过程。

网关（Gateway）

LoRa 网关位处 LoRa 星形网络的核心位置，是终端和服务器（Server）间的信息桥梁，是多信道的收发机。LoRa 网关有时又被称为 LoRa 基站或 LoRa 集中器，虽然定义不同，但其实是同一含义。

节点设备（Node / Device)

通过网关的信息转发，节点设备可以连接一个互联网上的应用，并将前端设备的数据上传到后台服务器。

注册（Registration)

用户可以注册节点设备或网关到云后台，在设备激活过程中可以校验设备的合法性，从而实现设备数据的传输。

私有云中间件

唯传科技提供的 lora 私有云中间件实现了标准 lorawan 节点和网关的协议对接和数据解析，并提供完善的基于 MQTT 接口和 rest 风格的 API 接口开放。用户不用关注前端硬件和传输协议的技术细节，和用户端直接打交道的将是 API 接口和 MQTT 协议，用户只要实现一个标准的 MQTT 客户端。唯传科技也为这些标准客户端和接口调用提供了源代码供参考。

云中间件服务

公有云的好处不断地吸引着企业用户们的关注，近年来其应用也有了显著的增长。但是在公有云大势所趋之下，还有一个企业更青睐的部署模式：虚拟私有云部署。唯传科技为了帮助客户解决物联网云服务的需求，同时考虑到客户独自部署云主机的技术困难，提供了一键部署服务。

1.4. 设计原则及特点

唯传科技为客户提供的云中间件服务的主要目的是希望采用先进的物联网技术，帮助传统企业能够快速实现产业转型和技术改造升级，为了实现这个目标，唯传科技的云中间件服务的设计原则主要体现在以下几个方面：

1、完全兼容 LORAWAN1.0 协议；
2、去中心化的网络建构，云平台不能只依赖某一个控制服务器；
3、高效地路由协议:最小的数据包和数据率带宽占用；
4、端到端的加密安全:应用层密匙保证应用层数据的安全；
5、专业的物联网基站技术：
很多物联网应用场景数据传输量小，并不需要大带宽，传统的 3G、4G 蜂窝式基站主要针对高速、大带宽应用，因此基站部署成本高，并不适合应用在物联网领域。唯传通过部署专门的低频段（1G HZ 以下）的物联网基站、结合唯传 SmartKit 物联网云平台，极大地弥补了这一缺陷。一个物联网基站同时可以支持多达数万个设备，信号可覆盖方圆数 10 万平方公里。
6、更低功耗、系统更可靠。
物联网应用很多是电池供电，需要数年的长时间供电，Airnode 技术正是基于 6LoWPAN 基础开发出来的一整套解决方案，6LoWPAN 是一种基于 IPv6 的低速无线个域网标准，支持低功耗，数据传输时节点设备才启动工作，
7、更开放、更兼容：
Airnode 标准协议包含各种无线技术，如 WiFi、蓝牙低功耗 BLE、RF 射频技术（868M/900M）、802.15.4、LoRa 等；
8、远距离传输
采用基于 LoRa 的传输技术，单节点可传输 3~20 公里距离；

由于采用了以上设计原则，因此，唯传科技的云中间件服务具有以下显著特点：

系统灵活性高

唯传科技的私有云中间件架构上是兼容了国际标准 lorawan 协议，遵循了该协议开放、灵活的设计原则，可以适应各种物联网应用。该中间件的设计原则是既可以兼容各种不同的软、硬件模块，也可以为用户做深度化和个性化定制，并提供全套的解决方案。

系统功能强大

唯传私有云中间件的设计兼容了LORAWAN标准协议，如入网、用户及设备授权管理、自适应码率控制、跳频、故障诊断等功能。因此该中间件模块可以为用户提供非常丰富的应用需求和适应各种不同的应用环境。

私有云中间件平台可以为不同的设备提供服务，如可以获取采集节点终端的传感器数据，也可以通过采用REST风格的API接口及MQTT接口为web应用端提供服务，同时还可以保证安全的数据访问。

系统稳定性强

唯传私有云中间件由于采用非常灵活的模块化设计原则，因此对各种复杂的功能需求及服务可以提供模块组合服务。这种服务模式可以非常易于测试及系统维护，这些服务已经为用户提供非常标准的API接口。在各种应用场景和仿真测试的数据都验证了系统稳定性能优良。

系统可复用性高

由于唯传私有云中间件采用了标准的MQTT协议与外部通信，因此系统具有很强的通用性，MQTT协议是一个非常好的轻量级协议，也可以封装成容易理解的其他接口类型，如REST API接口，用户不需要专业的指导就可以在此接口上开发出满足自己需求的各种应用。唯传科技提供的中间件服务可以无缝地与第三方平台对接，也可以直接集成到第三方的应用平台中，用户不用了解中间件中的处理技术细节，通过唯传科技中间件提供的订阅、发布服务，可以灵活地根据需求实现前端传感器节点数据的采集和命令控制下发。对用户而言，唯传私有云中间件就相当于一个透明的连接服务模块，在应用上层实现消息的传递和数据可视化展示。

用户开发更高效

由于唯传私有云中间件中集成了 MQTT，为用户提供非常方便的订阅 / 发布机制，因此用户可以在此基础上扩展其他的功能应用。在协议设计上，为了便于客户更好的理解和开发，我们也专门针对各单元模块优化了相关数据协议，使得用户也可以灵活的根据需要扩展自己的协议或架构，保证用户自己系统的通信带宽和能耗最优。

系统部署灵活

唯传私有云中间件由于采用分布式架构设计，因此云服务可以部署在不同的地方或多个公有或私有主机服务器上，不同的云服务主机可以互相协同工作。同时也可以部署在同一台计算机中提供本地化的设备服务。

1.5. LoRa 物联网系统组成

采用LoRa 技术的基于MQTT协议的消息服务系统架构组成如下所示，主要包含以下几个组成部分:

- **LoRa**节点**(LON):**

LoRa物联网系统的前端组成部分，各种接入传感器可以和区域内的 LoRa 网关上报数据，可以向网关发送和接收广播信号，采用 AES 加密保证数据的安全传输。

- **LoRa** 网关**(LGW):**

主要负责节点数据的采集和数据的转发给 LoRa 网络服务器，负载数 据格式采用 JASON 格式。

- **LoRa**网络服务器**LNS**(LoRa Network server):

网络服务器支持一个基本的 MQTT 服务器 MQB 交互，如将节点的数据暴露给 MQB 并接收控制命令回传给节点。

- **MQTT** 服务器 **MQB**(MQTT broker):

可以收集 MQTT 的发布/订阅命令并发布数据，提供了标准的授权机制使得只有授权的用户才可以访问数据资源。

- 应用服务器 **AS**（Application server）:

订阅某个网络服务器发布的各种资源，采用分布 式管理，节点设备数据可以通过 MQTT 协议经由 MQB 服务器使得这些数据可以被各种 应用服务器访问。每个节点设备可以通过标准的 API 接口与不同的应用交互。这些 API 接口可以是基于 JSON数据的MQTT 协议标准。

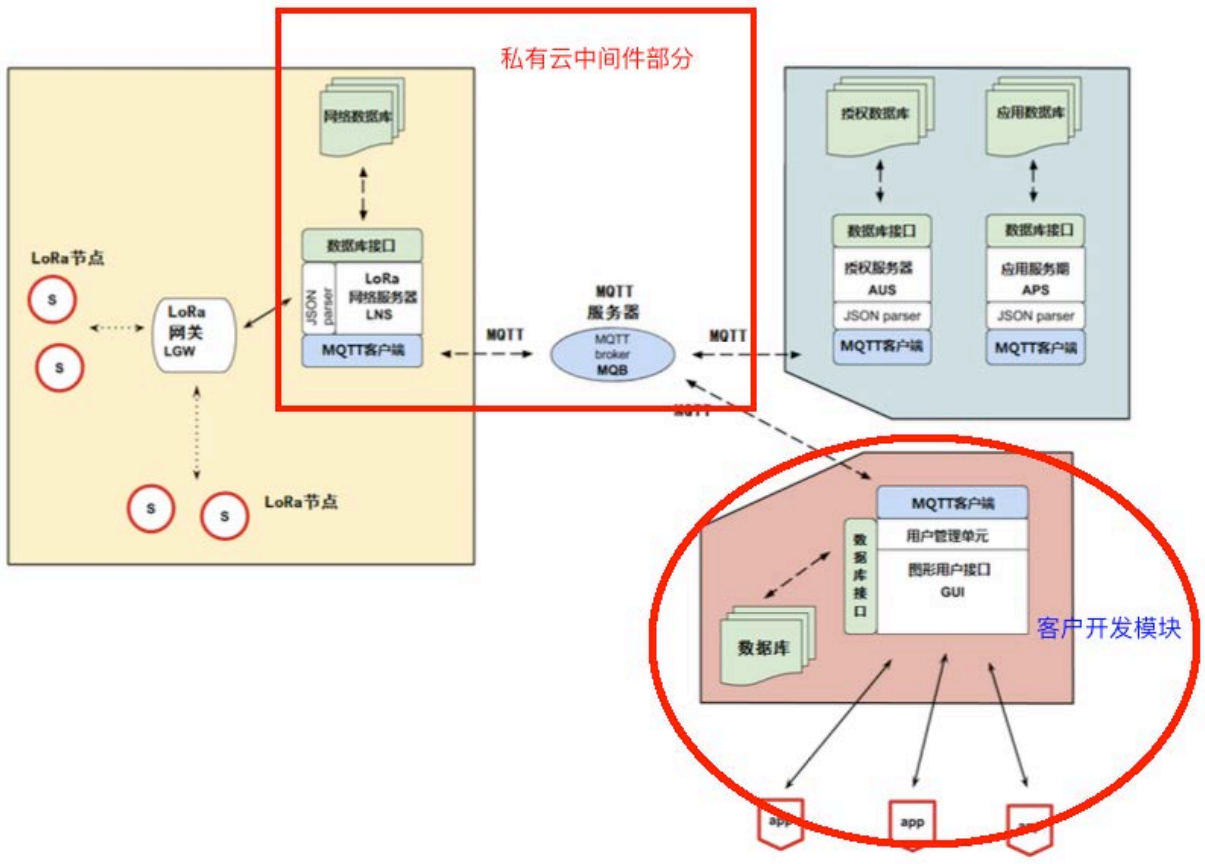
- 授权服务器 **AU**（Authorization server）:

负责管理不同的用户访问网络各资源的权 限，授权及前端节点设备的入网过程主要包含：**□每个用□**设置读/写/删除的权限；**接受□点□□的入网□求和会□密□、□算信息完整性加密**MIC，管理可能的冲突。

- 图形后台管理系统:

系统主要是基于Web的图形界面，可以方便用户通过图形界面与前端节点交互。管理用户可以将相关的配置操作通过 MQTT 协议实现对前端网关、节点设备的操控。

以下所示是基于LoRa的物联网系统组成框图，图中矩形框表示的是唯传科技提供的中间件模块，椭圆形框表示的是用户自己开发的系统。各系统之间主要是通过MQTT协议进行交互，此时用户只需要自己实现一个MQTT客户端的接口，不需要关注前端硬件和云平台交互的技术细节。



1.6. 云中间件服务架构

唯传科技提供的云中间件服务能够处理从网络中接收到来自网关的消息，对网关传上来的数据进行解密、数据格式化、存储。实现对前端终端设备的控制和数据收集。可以实现终端 ABP 和 OTAA 的注册模式，并且有提供给第三方调用的 API 接口，设置相关参数。提供 MQTT 接口，能够通过 MQTT 把终端注册信息

通知给 AS 应用服务器。

中间件系统框图如下：

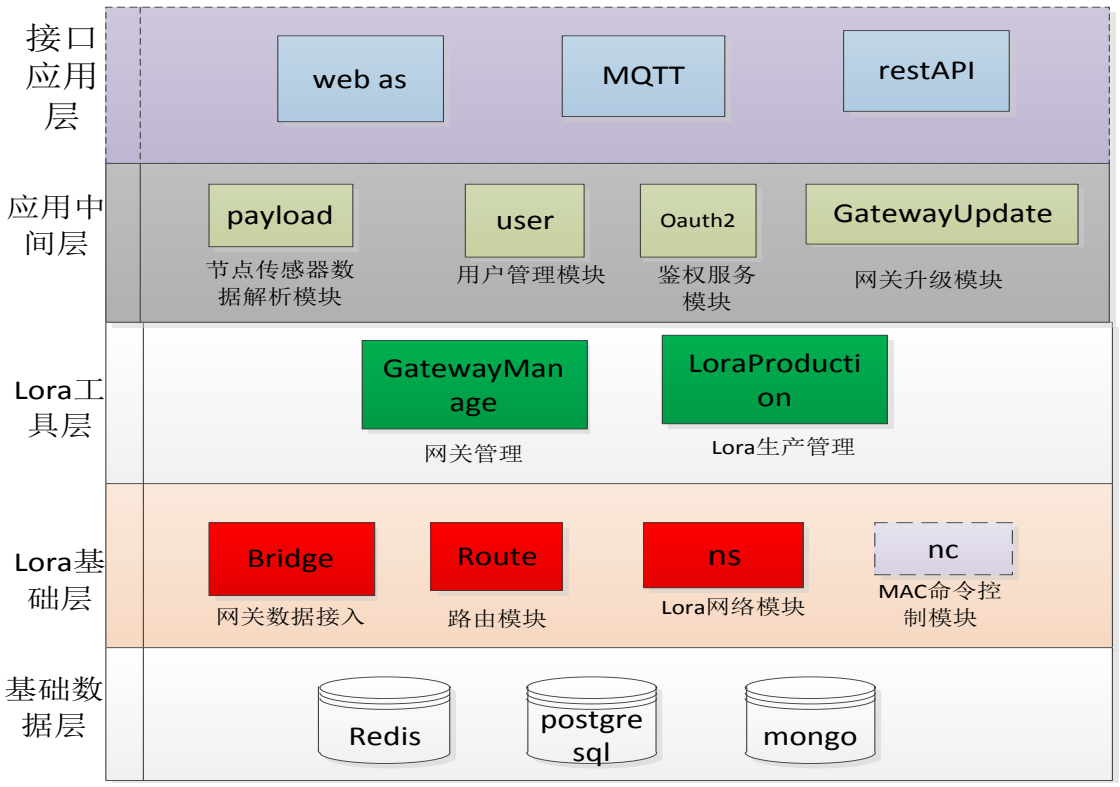


Figure 3-1 lora 中间件系统框图

如图消息中间件按应用划分， 由 5 层组成，每层由若干可独立运行模块提供服务。其中 Lora 基础层，为核心层，Bridge 模块、Route 模块、NS 模块可组网成最小系统。

基础数据层

基础数据层，提供数据保存服务：

- Redis 用于数据缓存
- postgresQL 用于设备的管理
- mongo 用户节点上传数据的保存，提供历史查询

Lora 基础层

该层为核心层 其中 Bridge 模块、Route 模块、NS 模块可组网成最小系统，NC 控制模块为可选，用于标准 MAC 命令处理。

该层提供建立 session、数据传输加密、路由功能、设备入网、注册网关、注册节点、MAC 命令上传下发，以及命令配置。

- Bridge 模块 设备接入模块
- Route 模块 用于对节点通讯频点进行频段的路由
- NS 模块 lora 网络处理模块，处理 lorawan 协议
- NC 模块 mac 命令控制模块

Lora 工具层

主要是为了测试、生产、以及网关和节点的调试使用，后期有需要可以开放给客户

- Gateway manage：网关管理用于展示网关上传信息，以及下发
- Lora Production：用于生产部管理测试出厂网关或者节点的平台，可以导出出厂网关或者节点。

应用中间层

主要提供应用数据的支持，非 lora 部分，其中 payload 的模块主要用于传感器数据的解析

- payload 解析用户定义以及我司定义的传感器数据保存到日志，并提供查询
- user 用户管理模块，提供企业用户管理，和个人用户管理包括用户注册、用户登录、用户授权、注销用户、删除用户
- oauth2 鉴权模块，提供令牌管理
- gateway Update 网关远程升级模块，提供网关的远程升级

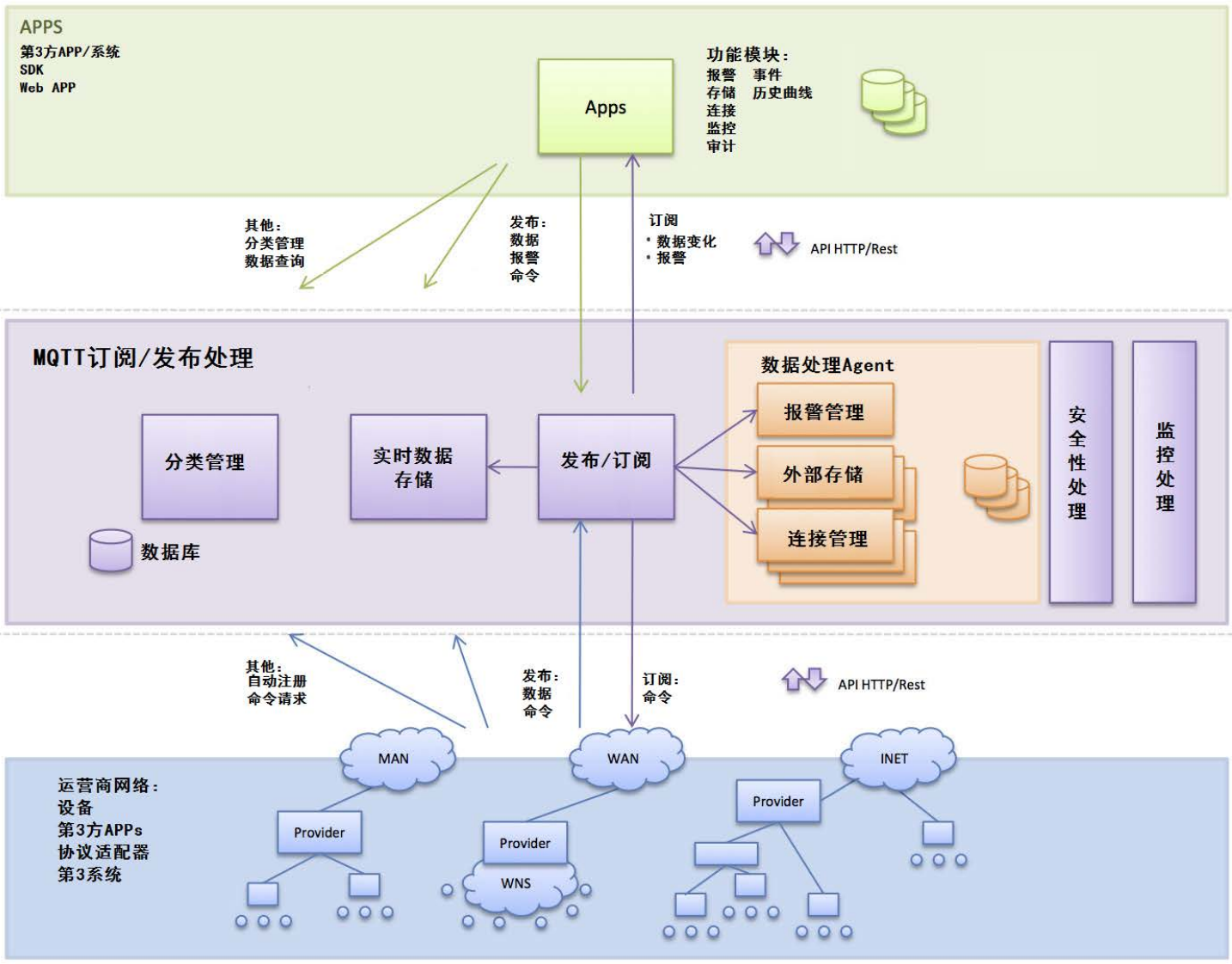
接口应用层

- Web AS 提供基于 lora 的 WEB UI 用于管理网关节点，以及展示节点上报信息
- MQTT 消息中间件，用于向用户提供网关、节点上报信息
- Rest API 接口用于向用户提供网关、节点的管理接口

1.7. 云服务系统架构

唯传物联网系统设计的原则是采用开放的标准 API 接口可以方便地实现与第 3 方平台及应用对接，以下是唯传物联网平台与第 3 方应用平台及运营商平台的交互示意图。

唯传私有云中间件服务采用了标准的 MQTT 协议与外部通信，因此系统具有很强的通用性，MQTT 协议是一个非常好的轻量级协议，同时我们也提供了基于 REST 风格的 API 接口供第三方调用，因此，唯传科技提供的中间件服务可以无缝地与第三方平台 APP 或运营平台对接，也可以直接集成到第三方的应用平台中，用户不用了解中间件中的处理技术细节，通过唯传科技中间件提供的订阅、发布服务，可以灵活地根据需求实现各种物联网应用。


APPS
第3方APP/系统
SDK
Web APP
Apps
功能模块：
报警 事件
存储 历史曲线
连接
监控
审计
其他：
分类管理
数据查询
发布：
数据
报警
命令
订阅
· 数据变化
· 报警
API HTTP/Rest
MQTT订阅/发布处理
分类管理
实时数据存储
发布/订阅
数据处理Agent
报警管理
外部存储
连接管理
安全性处理
监控处理
数据库
其他：
自动注册
命令请求
发布：
数据
命令
订阅：
命令
API HTTP/Rest
运营商网络：
设备
第3方APPs
协议适配器
第3系统
MAN
WAN
INET
Provider
Provider
Provider
WNS

API 说明

API 接口按照功能来分，主要可以分为：
应用管理 API 接口；
数据获取 API 接口；
Downlink 下发 API 接口；

1.8. 应用管理

通过应用管理接口，用户能够注册 APP 应用、网关、节点，需要注意的是，在注册网关和节点的时候需要用到网关和节点出厂时的授权码，以判断网关、节点的合法性。否则注册失败。该接口可以通过 restfull API 接口或者 gRPC 接口调用。

管理 API 调用流程：

1、 注册 APP 应用，需要注意在注册时 APPEUI 需要和节点保存的 APPEUI 一致，节点在出厂时默认 APPEUI 为 0000000000000001；
2、 注册网关；
3、 注册节点；

1.9. 数据获取

通过该接口，能够接收到节点上传的信息。该接口通过 MQTT 实现，需要订阅所关心节点的主题。实际例子请参考：MQTT 举例章节

数据传输如下图所示：

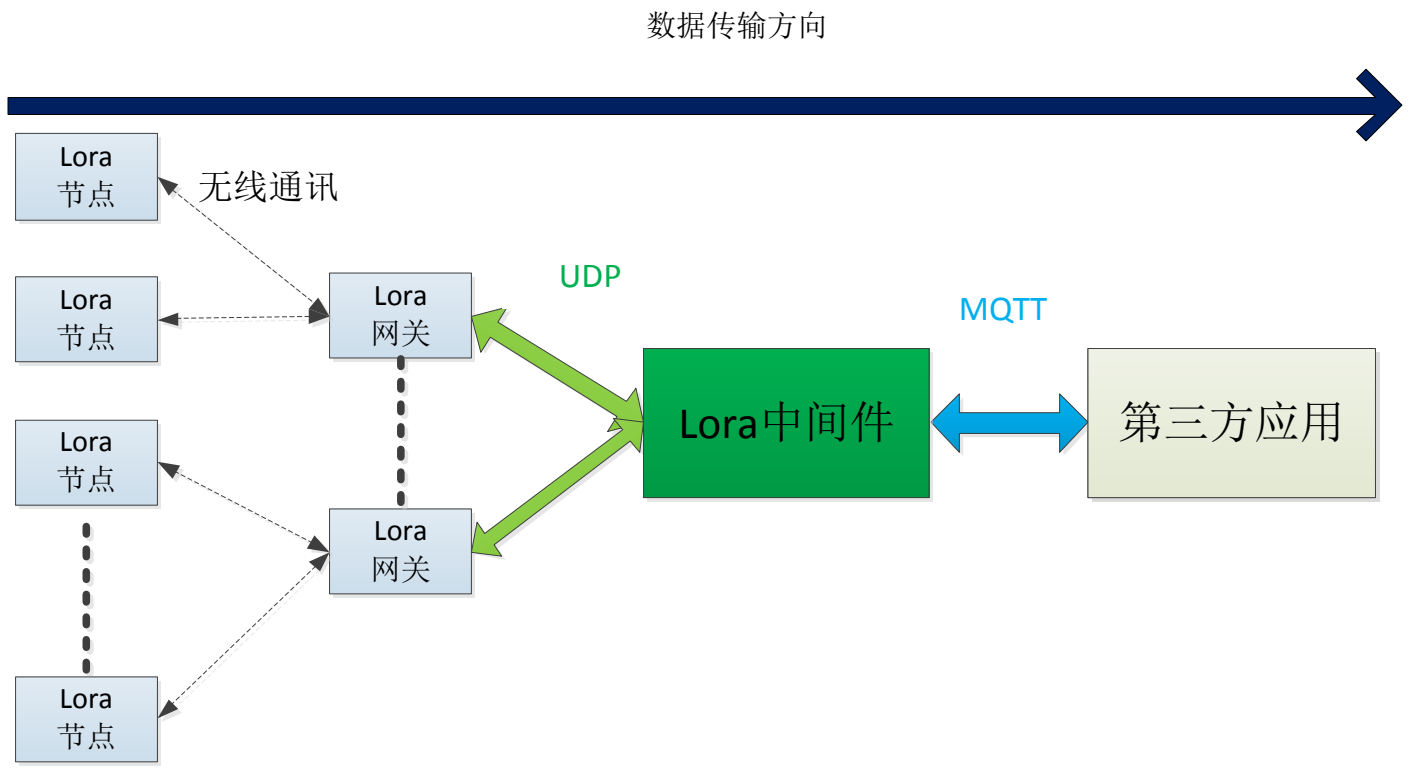


Figure 4-1 uplink 数据流向图

1.10. 命令下发

该 API 接口支持用户下发命令给节点，使节点执行相应的操作。在 V1.0 版本中改接口由 MQTT 消息的方式提供，在 V2.0 版本中由 restful API 接口提供。

该数据流程图如下：

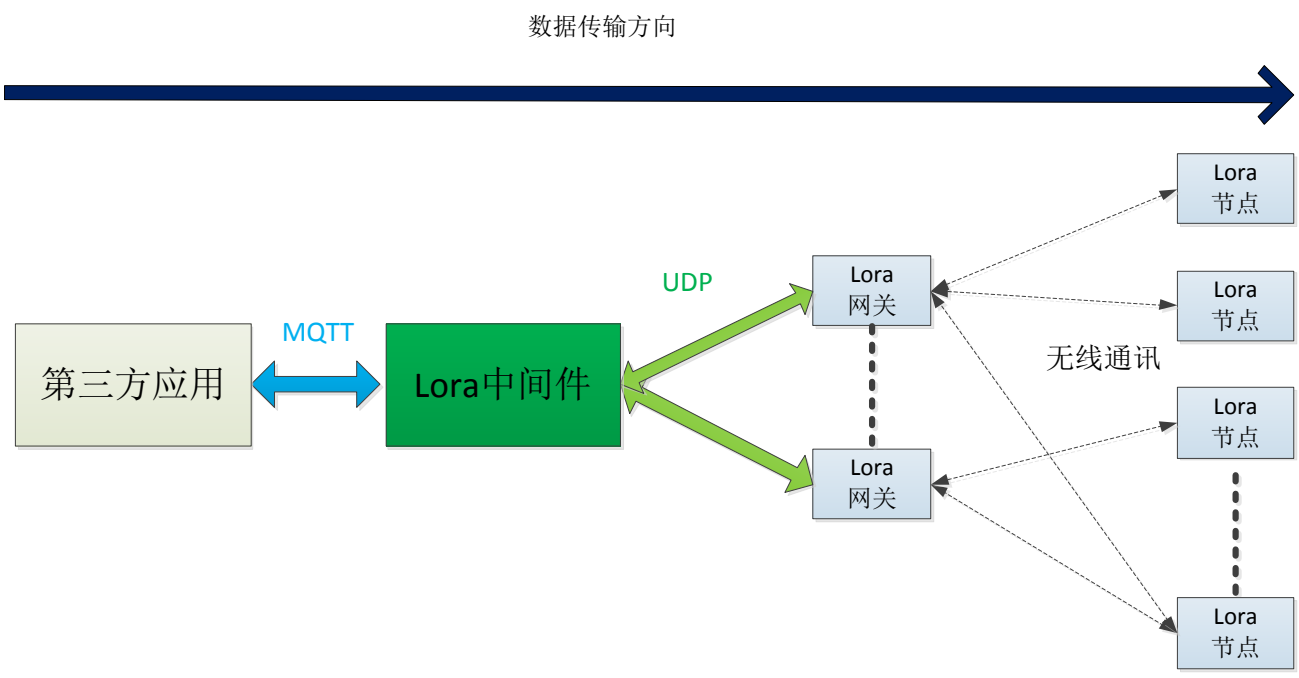


Figure 4-2 downlink 数据流向图

1.11. 安全性

唯传科技提供的云中间件服务支持端到端的数据加密，由于对每个设备都采用了128位的密匙加密，因此可以有效防止攻击，同时基于LORAWAN协议的传输协议实现了3层加密机制， 数据在传输过程中无缝被第三方捕获和破解，保证了客户数据的高度安全性。

1.12. MQTT 及 MQTT 客户端

MQTT 是一个轻量级的物联网协议，方便提供基于发布 / 订阅机制的数据传输协议。
您可以从互联网上找到很多基于 MQTT 协议实现的 MQTT 客户端，其中比较有名的包括：
Eclips 的开源项目 Paho：
http://www.eclipse.org/paho/
Mosquitto 也提供了 CLI 工具实现 MQTT 消息的订阅和发布机制。
https://mosquitto.org
MQTTBox 也是一个非常好的图形 GUI 客户端。
http://workswithweb.com/mqttbox.html

唯传科技也为客户提供了客户端 SDK 包示例，供客户参考学习，将支持 Go、Java、Node.js，目前 Node.js 包已经提供，其他语言包后续提供，参考 SDK 包源码，使之调用 API 接口，更加简单。

操作实例

1.13. 连接网关

云中间件网关注册

管理配置地址：http://xxx.xxx.xxx.xxx:8000
xxx.xxx.xxx.xxx 为服务器 IP 地址
在平台注册网关时需要用到我司的网关授权码，授权码在网关出厂时会分配。
如网关 ID: 0002e4956e405b8d ，授权码：c8bf20d77500f666badf6414d9bd8bxx
在平台管理页面注册网关，如下图所示：

Figure 5-1 创建网关

网关配置

首先需要配置你的网关，确保你的网关连接的服务器端口和 IP 地址正确。
进入网关的配置页面，如下图：

Figure 5-2 lora 网关设置

上图中，服务器地址要设置成云中间件部署的地址，上行端口和下行端口要设置和服务器一致，默认端口号为 1680

- **检查是否正确连接**

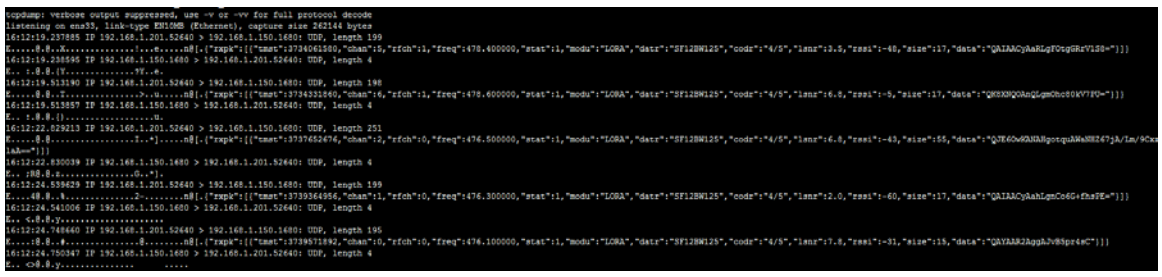
  确定网关外围正确安装，请参考《网关使用说明》

- **检查端口是否有收到数据包**

  在服务器上监听1680端口，在控制台执行命令：

```
sudo tcpdump -AUq port 1700
```

  如果网关正确连接，并且运行，则会看到如下信息：



- 检查网关日志

通过网关UI页面可以查看网关状态，截图如下：

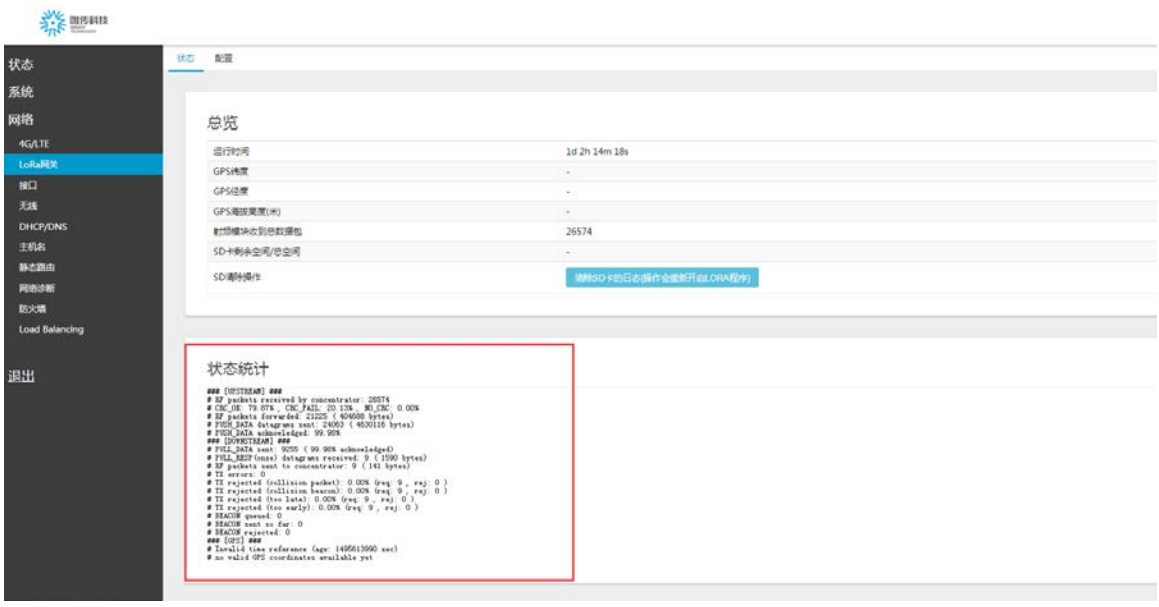


Figure 5-3 网关日志统计

1.14. 建立节点通信

云中间件节点注册

节点注册和网关注册一样也需要授权码，该授权码，在节点出厂时由我司分配。

例如，节点：ffffff000002bd40，授权码：7468a09d9f5aee45b4e1a5b2010439XX

如下图：

Figure 5-4 节点注册

注意：红框部分是必须要填的（其他的采用默认值即可），APPEUI、APPKEY 默认值如下：

APPEUI	0000000000000001
APPKEY	98929b92f09e2daf676d646d0f61d250

节点配置

节点的 APPEUI、APPKEY 和节点 ID 需要和节点在云中间件注册时一致，如果采取默认值，则不需要修改，直接上电即可。

OTAA 方式入网，需要配置 AppEui、AppKey；ABP 入网需要配置 NetID，DevAddr，NwkSKey，AppSKey。频点需要与网关一致。

1.15. 管理后台数据解析

节点上传数据必须符合 lorawan 规范，因此节点上传数据填充在 FRMPayload 数据帧里。

FRMPayload 里的数据格式根据 LoRaWAN 协议的 FPort 号来决定。因为节点入网后的每条 LoRaWAN 消息都会带有 FPort 字段，所以可以使用 FPort 字段的值来区分 FRMPayload 里的数据格式。其中可使用的范围 1～222，如果以后类型超过 222 个，可以使用 223 这个 FPort 号来表示大于 222 的类型数据，当 FPort 等于 223 时在 Payload 里的前两个字节来表示大于 222 的 FPort，后面对应相应的数据格式。224～255 为 LoRaWAN 协议保留。FPort 为 1 专用于中继转发。详细规则如下：

Fport	UP/DOWN	Payload 格式	描述/备注
0	UP/DOWN	Mac Command	Mac 命令专用
1	UP	DevEUI+Port+Payload	该 Port 号用来做中继转发专用，DevEUI：8Byte，Port：1Byte

Fport	UP/DOWN	Payload 格式	描述/备注
2	UP	井盖监测报警	占用 3 个字节，第 1 个字节，0：井盖未打开，1：井盖打开，其它字节默认为 0。节点发送待确认的数据帧，直到收到服务器响应的 ACK
3	DOWN	GPIO_OUT_0、 GPIO_OUT_1、 GPIO_OUT_2… GPIO_OUT_N	一个字节对应一个 GPIO 输出，从 GPIO 0 开始，1：表示高电平，0：表示低电平。长度由所使用的频段和速率决定。
4	UP	温度+湿度	℃ * 100（singned 16 bits） %RH * 100（unsigned 16 bits），payload 中按大端格式填充。值为 0xFFFF 时表示获取传感器数据失败。服务器显示数据需要除以 100。
5	UP	GPS 数据： 纬度（3Byte）+ 经度（3Byte）+ 高度（2Byte）	GPS 经纬度按如下编码格式编码：纬度使用有符号 24 位表示，当值为-2^{23}对应南纬 90°；当值为 2^{23}对应北纬 90°；值 0 表示赤道。经度使用有符号 24 位表示，当值为-2^{23}对应西经 180°；当值为 2^{23}对应东经 180°；值 0 表示格林威治子午线。高度使用有符号 16 位表示，单位米。
6	UP	温度	℃ * 100（singned 16 bits）payload 中按大端格式填充。值为 0xFFFF 时表示获取传感器数据失败。服务器显示数据需要除以 100。
7	UP	ESD500 Modbus 读寄存器地址为 0～4 的 13 字节应答帧	第一个字节为应答的设备地址，第二个字节为功能码，第三个字节为数据长度，第四字节开始为应答数据：寄存器 0～4 的值，一个寄存器为 2 字节(有符号)。
8	UP	6 字节水表编号+4 字节水表读数+2 字节水表状态码	资江水表；水表编号和读数采用 BCD 编码，水表的读数第 4 字节为小数部分。水表状态码第二个字节为保留值，查看第一个字节。第一字节对应的含义：Bit 0 存储器状态 (1:故障,0:正常)；Bit 1 阀门状态 (1:故障,0:正常)； Bit 2 信号状态 (1:故障,0:正常)； Bit 3 电池状态 (1:故障,0:正常)； Bit 4 保留； Bit 5 保留； Bit 6 水表通讯状态（1:故障,0:正常）；Bit 7 阀门开关状态 (1:开,0:合)；
9	UP	空气质量指数（PM2.5 AQI）	共两个字节，无符号 16 位整型（小端模式）
10	UP	电池电量百分比	占用 1 个字节，电池电量百分比：0%~100%
11	UP	红外报警	占用 1 个字节，0：无报警，1：报警。节点发送待确认的数据帧，直到收到服务器响应的 ACK
12	UP	地磁报警（车辆检测）	占用 1 个字节，0：未检测倒车辆，1：检测倒车辆。节点发送待确认的数据帧，直到收到服务器响应的 ACK
13	UP	GPS 经纬度数据	纬度使用有符号 24 位表示，当值为-2^{23}对应南纬 90°；当值为 2^{23}对应北纬 90°；值 0 表示赤道。经度使用有符号 24 位表示，当值为-2^{23}对应西经 180°；当值为 2^{23}对应东经 180°；值 0 表示格林威治子午线。节点一次会发送 1~8 次采样的 GPS 数据，每次采样的数据均由纬度（3Byte）和经度（3Byte）组成
14	UP	电池剩余电量低于 20%报警	占用 1 个字节，0：剩余电量高于 20%，1：剩余电量低于 20%。节点发送待确认的数据帧，直到收到服务器响应的 ACK
15	DOWN	丢失模式设置	占用 1 个字节，0：解除丢失模式，1：启动丢失模式。服务器下发待确认的数据帧，直到收到节点响应的 ACK
16	UP	PH 值	占用 2 个字节，无符号 16 位整型（高字节在前），显示实际 PH 值时需要除以 100
17	UP	一个字节的电流状态值和 4 个字节的交流变送器的电流值，单位 mA，无符号 32 位整型	对于第一位的电流状态值按 16 进制显示，后面的电流值为交流电流变送器的值 x100，单位 mA。后台显示时需要除以 100 后才是传感器的真实值。
18	UP	中继及内网节点状态上报	占用 1~49 个字节，第 1 个字节为中继电源状态，0：电池电压正常，1：电池电压过低；第 2~49 字节平均分为三组，每组 16 个字节，其中第 1~8 字节为 DevEUI，第 9~12 字节为实际接收到的数据包个数，第 13~16 字节为应收到的数据包个数，用于统计内网节点的丢包率。其余字节，依此类推。

Fport	UP/DOWN	Payload 格式	描述/备注
19	UP	烟感报警	占用 1 个字节，0：无报警，1：报警。节点发送待确认的数据帧，直到收到服务器响应的 ACK
20	UP	门磁报警	占用 1 个字节，0：无报警，1：报警。节点发送待确认的数据帧，直到收到服务器响应的 ACK
21	UP	浮球液位控制器报警	占用 2 个字节，第 1 个字节，0：不需要排水，1：需要排水；第 2 个字节，0：不需要供水，1：需要供水。节点发送待确认的数据帧，直到收到服务器响应的 ACK
22	UP	电接点压力表报警	占用 2 个字节，第 1 个字节，0：正常范围内，1：压力高于上限；第 2 个字节，0：正常范围内，1：压力低于下限。节点发送待确认的数据帧，直到收到服务器响应的 ACK
23	UP	FLOW SWITCH 报警+水浸传感器	占用 2 个字节，第 1 个字节，0：无水流，1：有水流；第 2 个字节，0：未被水浸，1：水浸。节点发送待确认的数据帧，直到收到服务器响应的 ACK
24	UP	温湿度变送记录仪	占用 4 个字节，第 1、2 字节表示湿值度（16 位无符号整型），第 3、4 字节表示表示温度值（16 位有符号整型），高位在前，实际值需除以 10。如 0x02040109，表示湿度 51.6%，温度 26.5℃
25	UP	水浸传感器 X2	占用 2 个字节，第 1 个字节（溢水检测），0：未被水浸，1：水浸；第 2 个字节（无水检测），0：未被水浸，1：水浸。节点发送待确认的数据帧，直到收到服务器响应的 ACK
25~220	UP	RFU	RFU
221	UP/DOWN	UpLink 丢包率	显示数据时除以 100
222	UP/DOWN	RS232、RS485 数据透传	SF12BW125K 时：服务器下发数据长度限制：46 字节。节点上报数据长度限制：51 字节，其他速率请查阅 LoRaWAN 白皮书
223	UP/DOWN	Port+Payload	FPort 扩展专用，Port：2Byte
224	UP/DOWN	自行定义	测试专用
225~255	UP/DOWN	RFU	RFU

Table 5-1 节点 payload 表格

注：UP/DOWN：上行/下行

1.16. MQTT 举例

本示例可以让您理解如何采用Mosquitto工具订阅一个消息及对特定设备发送响应。对于Mosquitto不了解的用户，可以访问：https://mosquitto.org/，了解详细的使用方法。
首先需要下载Mosquitto工具，https://mosquitto.org/download/
在安装 Mosquitto 服务时，会提供以下工具：

- Mosquito_sub :MQTT 消息订阅工具；
- Mosquito_pub：MQTT 消息发布工具；

云中间件提供以下主题订阅：

节点入网上报

订阅主题：application/[AppEUI]/node/[DevEUI]/join

- APPEUI：需要与注册节点时的 APPEUI 保持一致；
- DevEUI：节点 ID，需要与注册节点的 ID 保持一致；

返回消息格式：

```
{
    "devAddr": "06682ea2",          // 节点地址
```

```
    "DevEUI": "0202020202020202"  // 节点 EUI
}
```

例如，节点 ID：ffffff000002bd40，注册时 APPEUI: 0000000000000001

使用 MQTT 消息订阅工具，命令如下：

```
mosquitto_sub    -u user -P password -t application/0000000000000001/node/ffffff000002bd40/join
```

其中： user 为 MQTT 配置的用户名称（如果没有可以用设置）

Password 为 MQTT 配置的密码（如果没有可以用设置）

当节点发起入网申请后，会收到如下信息：

```
{"devAddr":"03fd63da","devEUI":" ffffff000002bd40"}
```

节点数据上报

订阅主题：application/[AppEUI]/node/[DevEUI]/rx

- APPEUI：需要与注册节点时的 APPEUI 保持一致；
- DevEUI：节点 ID，需要与注册节点的 ID 保持一致；

返回消息格式：

```
{
    "devEUI": "0202020202020202",  // 节点 EUI
    "fPort": 5,                    // FPort 命令
    "gatewayCount": 3,              // 网关数量
    "rssi": -59,                   // 信号强度
    "data": "..."                     // base64 编码 payload
}
```

例如，节点 ID：ffffff000002bd40，注册时 APPEUI: 0000000000000001

使用 MQTT 消息订阅工具，命令如下：

```
mosquitto_sub    -u user -P password -t application/0000000000000001/node/ffffff000002bd40/rx
```

其中： user 为 MQTT 配置的用户名称（如果没有可以用设置）

Password 为 MQTT 配置的密码（如果没有可以用设置）

当节点发起入网申请后，会收到如下信息：

```
{
    "devEUI":" ffffff000002bd40",
    "time":"0001-01-01T00:00:00Z",
    "fPort":224,
    "gatewayCount":1,
    "rssi":-2,
    "data":"AAAeGw=="
}
```

节点状态上报

该主题主要反馈节点的通信状态

订阅主题：application/[AppEUI]/node/[DevEUI]/rxinfo

- APPEUI：需要与注册节点时的 APPEUI 保持一致；
- DevEUI：节点 ID，需要与注册节点的 ID 保持一致；

返回消息格式：

```
{
    "devEUI": "0202020202020202",
    "adr": false,
    "fCnt": 1,
    "rxInfo": [{
        "mac": "1dee08d0b691d149",
        "time": "2016-05-01T10:50:54.973189Z",
        "timestamp": 1794488451,
        "frequency": 868100000,
        "channel": 0,
        "rfChain": 1,
        "crcStatus": 1,
        "codeRate": "4/5",
        "rssi": -54,
        "loRaSNR": 10.2,
        "size": 17,
        "dataRate": {
            "modulation": "LORA",
            "spreadFactor": 7,
            "bandwidth": 125
        }
    }]
}
```

例如，节点 ID：ffffff000002bd40，注册时 APPEUI: 0000000000000001
使用 MQTT 消息订阅工具，命令如下：

```
mosquitto_sub    -u user -P password -t application/0000000000000001/node/ffffff000002bd40/rxinfo
```

其中： user 为 MQTT 配置的用户名称（如果没有可以用设置）
Password 为 MQTT 配置的密码（如果没有可以用设置）

当节点发起入网申请后，会收到如下信息：

```
{
    "devEUI":"ffffff00000000ea",
    "adr":true,
    "fCnt":1110,
    "rxInfo":[{
            "mac":"0002e4956e405b8d",
            "time":"0001-01-01T00:00:00Z",
            "timestamp":3701779820,
            "frequency":476300000,
            "channel":1,
```

```
            "rfChain":0,
            "crcStatus":1,
            "codeRate":"4/5",
            "rssi":-1,
            "loRaSNR":7,
            "size":17,
            "dataRate":{
                  "modulation":"LORA",
                  "spreadFactor":12,
                  "bandwidth":125
             }
     }]
}
```

采用 Node.js 客户端监听 MQTT 消息示例

以下是一个利用 node.js 客户端实现的 MQTT 消息监听示例：

1.下载 MQTT 客户端包，这里选择对应的语言下载 https://github.com/mqtt/mqtt.github.io/wiki/libraries。本系统使用的是 Node.js ，使用命令行安装 npm install mqtt --save

2.建立连接，监听

```
var mqtt = require('mqtt');
//用户名+密码+地址+端口号，示例   mqtt://user:password@127.0.0.1:1883
var mqttUrl = 'mqtt://'+user+':'+password+'@'+host+':'+port;
var mqttClient = mqtt.connect(mqttUrl);
mqttClient.on('connect', function () {
       client.subscribe('application/#');
       client.subscribe('gateway/#');
})
mqttClient.on('message', function (topic, data) {
      console.log(topic, data);
})
mqttClient.on('error', function (err) {
       console.error('mqtt error:',err)
})
```

3.上面就完成的监听，收到信息可以进行存储（本系统用的是 MongoDB 存储这些消息）。这样就可以根据这些数据做一些图表分析、地图定位、监控等应用。消息数据的详细类型和格式可以参考 MQTT 消息文档。

1.17. API 调用举例

API 遵循 RESTful API 的标准，使用 HTTP 的协议，以 JavaScript 为例，调用创建网关 API 如下：

```
$.ajax({
    dataType:"json",
    type:"post",//请求方法
    url:"/api/v1/gateway",//请求地址
    data:{
        "gatewayAddr": "00000000",
        "gatewayEUI": "000388c2558d942d",
        "gatewaySeq": "c8bf20d77500f666badf6414d9bd8b84"
    },//请求数据
    success:function(response){
        console.log(response);//成功执行
    },
    error:function(e){
        console.log(e);//失败执行
    }
})
```

请求成功后，返回结果如下：

```
{
    "Code": "200"
}
```

1.18. gRPC 调用举例

唯传科技为了帮助更好的实现跨平台开发，提供了非常友好的 RPC 调用接口 gRPC 工具，可以实现用户自己的特定应用，如帐户管理、网关设备信息查询等功能。

1.首先，安装 gRPC 工具包。官网有相关语言的包，http://www.grpc.io/docs/。由于本系统是基于 Node.js 开发，使用 npm install grpc --save 命令行下载 gRPC 包。

2.复制 gateway.proto 和 node.proto 文件到根目录，如没找到，请找相关人员。proto 文件详细说明请查看 https://developers.google.com/protocol-buffers/docs/proto3

3.实现代码

```
var grpc = require('grpc');
// grpc 配置连接
var proto = {
    gateway: grpc.load('./gateway.proto').api,
    node: grpc.load('./node.proto').api
}
var address = '中间件的地址+端口';
var gatewayClient = new proto.gateway.Gateway(address, grpc.credentials.createInsecure());
```

```
var nodeClient = new proto.node.Node(address, grpc.credentials.createInsecure());
```

4.上面建立了 gRPC 的连接，那么下面就来实现基本调用，必须查询网关信息

```
var gatewayEUI = '网关 EUI';
gatewayClient.get({
      gatewayEUI: gatewayEUI
},    function (err, data) {
      console.log(err, data)
})
```

就这样完成了查询网关信息，现在你可根据 gRPC 接口文档做更多的开发了。

1.19. 远程诊断及调试

唯传科技的云中间件服务的 Web 管理平台也提供了非常方便的远程诊断及调试工具，该工具让 LoRa 中间件维护人员能够通过调试工具快速定位问题所在，可调试的工具有获取节点状态、设置占空比、设置接收延时、设置接收参数、设置速率、创建修改通道的调试功能。

填充内容注释（重复的不举例）

设置占空比：

- appEUI 节点 appEUI
- devEUI 节点 devEUI
- maxDCCycle 最大占空比
- FRMPayload 是否放在 FRMPayload 中发送

设置接收延迟：

- delay 延迟指定时间间隔（秒）

设置接收参数：

- Frequency 接收窗口使用的信道频率
- rx1DROffset 设置终端设备上行和下行第一个接收窗口（RX1）数据速率之间的偏移
- rx2DataRate 定义第二个接收窗口使用的数据速率

创建修改通道：

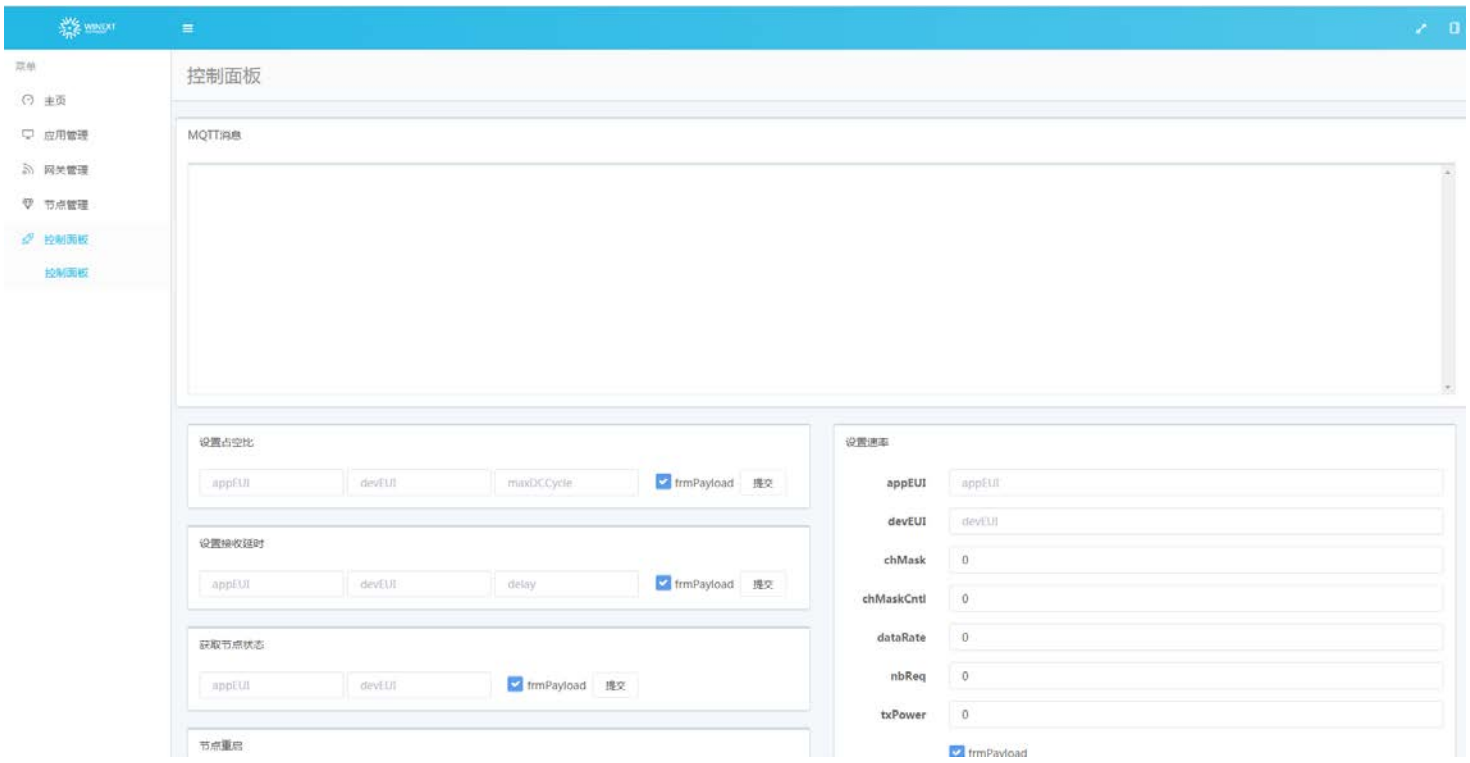
- chIndex 信道索引
- freq 信道频率
- maxDR 最大数据速率
- minDR 最小数据速率

设置速率：

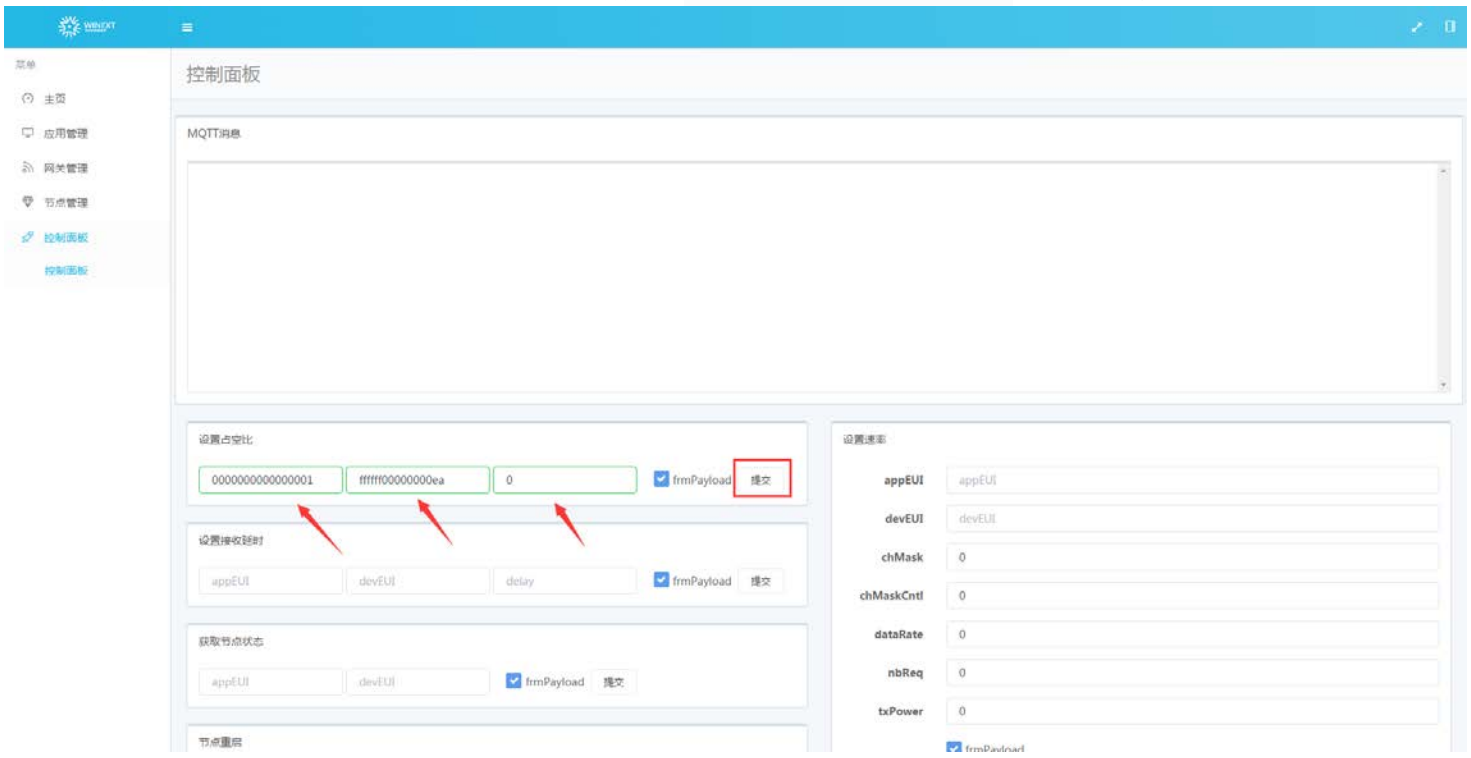
- chMask 信道掩码
- chMaskCntl 信道掩码控制
- dataRate 数据速率
- nbReq uplink 包重复次数
- txPower TX 输出功率

使用流程以设置占空比为例，其他功能调试方法同理，具体如下：

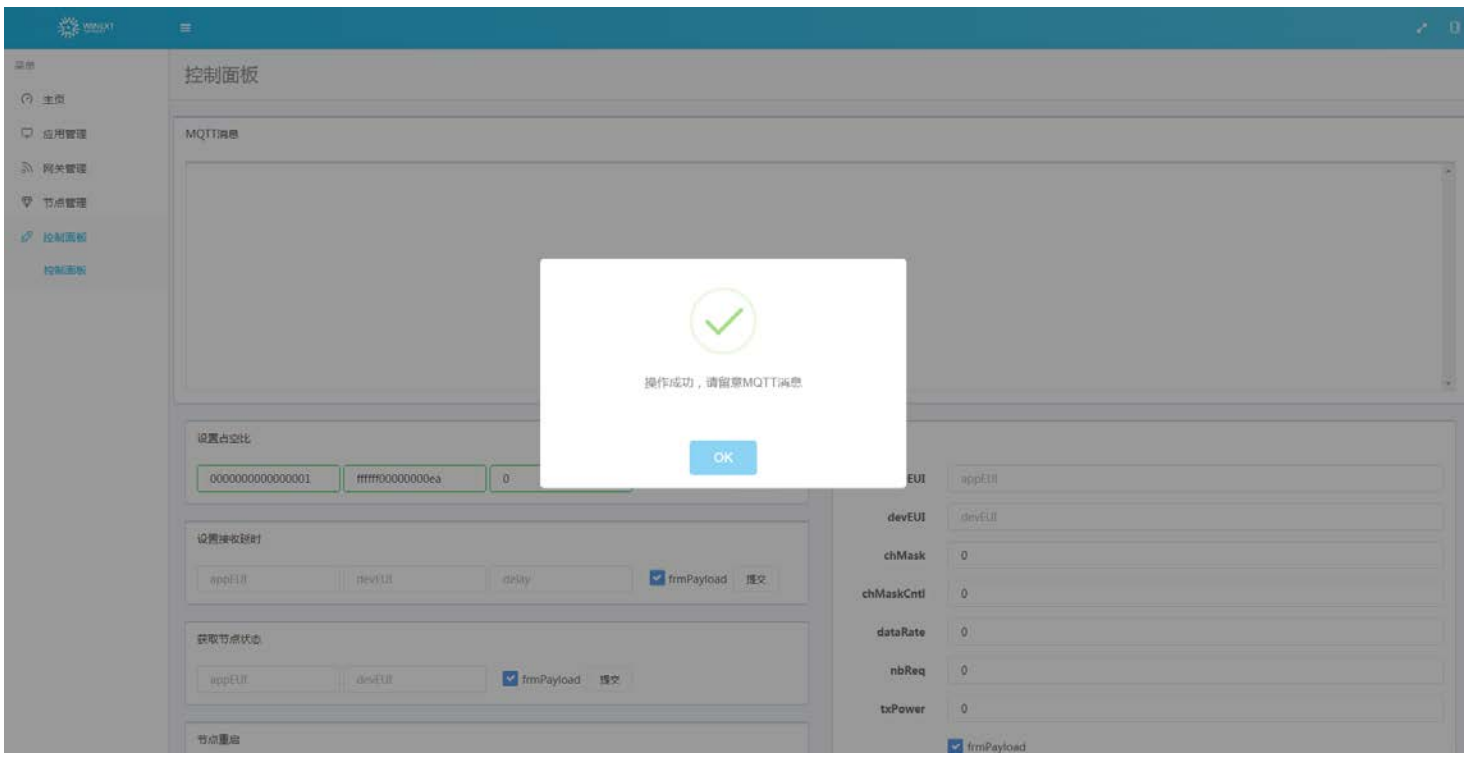
1） 打开 Web AS 后点击控制面板，界面如下：



2） 填充所需调试功能的对应表单（这里以节点 fffff00000000ea 为例），界面如下：



3） 填充好内容后，点击表单所对应的提交按钮，会得到如下的界面提示：



4） 点击 OK 按钮，留意 MQTT 消息返回

常见问题

LoRa 是什么？

LoRa 是低功耗广域网通信技术中的一种，是 Semtech 公司采用和推广的一种基于扩频技术的超远距离无线传输技术，是 Semtech 射频部分产生的一种独特的调制格式。LoRa 射频部分的核心芯片是 SX1276 和 SX1278。这类芯片集成规模小、效率高，为 LoRa 无线模块带来高接收灵敏度。而网关芯片则采用的是集成度更高、信道数更多的 SX1301。用 SX1301 作为核心开发出的 LoRa 网关，可以与许许多多的 LoRa 模块构成多节点的复杂的物联网自组网。从技术形态来看，LoRa 是一种扩频技术，但它不是直接序列扩频。直接序列扩频通过调制载波芯片来传输更多的频谱，从而提高编码增益。而 LoRa 调制与多状态 FSK 调制类似，使用未调制载波来进行线性调频，使能量分散到更广泛的频段。从网络结构来看，LoRa 调制技术本身是一个物理层（PHYlayer）协议，能被用在几乎所有的网络技术中。Mesh 网络虽然扩展了网络覆盖的范围，但是却牺牲了网络容量、同步开销、电池使用寿命。随着 LoRa 技术链路预算和覆盖距离的同时提升，Mesh 网络已不再适合，故采用星形的组网方式来优化网络结构、延长电池寿命、简化安装。LoRa 网关和模块间以星形网方式组网，而 LoRa 模块间理论上可以以点对点轮询的方式组网，当然点对点轮询效率要远远低于星形网。

LoRa 网关是什么？

LoRa 网关位处 LoRa 星形网络的核心位置，是终端和服务器（Server）间的信息桥梁，是多信道的收发机。LoRa 网关有时又被称为 LoRa 基站或 LoRa 集中器，虽然定义不同，但其实是同一含义。
LoRa 网关使用不同的扩频因子，不同的扩频因子两两正交因而理论上可以在同一信道中对多条不同扩频因子的信号进行解调。网关与网络服务器间通过标准 IP 进行连接，终端通过单跳与一个或多个网关进行通讯，所有的终端通讯都是双向通讯，同时也支持软件远程升级等。以下几个网关的几个重要的特点：
网关的分类：目前来说，定义不同，网关类型也不同。例如，按照应用场景不同可分为为室内型网关和室外型网关；按照通讯方式不同可分为全双工网关和半双工网关；而按照设计标准不同可分为完全符合 LoRaWAN 协议网关和不完全符合 LoRaWAN 协议网关。WINEXT 新一代网关为室外型，半双工，并且完全符合 LoRaWAN 协议。完全符合 LoRaWAN 协议的 LoRa 网关及 LoRa 终端能够实现互联互通，这具有很大意义！

网关的容量是如何的？

网关容量是指在一定时间内网关接收数据包数量的能力。理论上来说，单个 SX1301 芯片拥有 8 个信道，在完全符合 LoRaWAN 协议的情况下最多每天能接收 150 万个数据包。如果某应用发包频率为 1 包/小时，单个 SX1301 芯片构成的网关能接入 62500 个终端节点。当然，这只是一个理论值，网关接入终端数量最终还是与网关信道数量、终端发包频率、发包字节数和扩频因子息息相关。

网关接入点决定因素有哪些？

LoRa 网关接入的节点数取决于 LoRa 网关所能提供的信道资源以及单个 LoRa 终端占用的信道资源。LoRa 网关如果采用 Semtech 标准参考设计，网关采用 SX1301 芯片，那么信道数是固定的 8 个上行信道 1 个下行信道。物理信道数确定了， LoRa 网关所能提供的信道资源也就确定了。（网关设计相同，信道数相同，WINEXT 网关能实现 8 个上行，1 个下行。）单个 LoRa 终端占用的信道资源与终端占用信道的时间一致，也就与终端的发包频率、发包字节数以及 LoRa 终端的扩频因子息息相关。当 LoRa 终端的发包频率和发包字节数上升，该终端占据信道收发的时间就会增加，就占用了更多的信道资源。而当 LoRa 终端采用更大的扩频因子时，信号可以传的更远，但是代价是传递单位字节的信息会花费更多的时间。
除了网关外，我们也需要关注 LoRa 终端：LoRa 终端是 LoRa 网络的组成部分，一般由 LoRa 模块和传感器等器件组成。LoRa 终端可使用电池供电，能够远程定位。每一个符合 LoRaWAN 协议的终端都能与符合 LoRaWAN 的网关直接通讯，从而实现互联互通。

LoRa 的频段选择是如何定义的？

按理论来说，可以使用 150 MHz 到 1 GHz 频段中的任何频率。但是 Semtech 的 LoRa 芯片并不是所有的 sub-GHz 的频段都可以使用，在常用频段（如 433MHz，470MHz~510MHz，780MHz 以及欧美常用的 868MHz 和 915MHz 都属于常用频段）以外的一些频率并不能很好的支持。WINEXT 提供 470-510MHz、863-870MHz、902-928MHz 频段网关。

LoRa 网关使用免费频段，会不会容易受到频率干扰？

抗干扰能力取决于 LoRa 技术本身的特性和网关的设计。LoRa 技术本身拥有超高的接收灵敏度（RSSI）和超强信噪比（SNR）。以 WINEXT 的 LoRa 网关与 LoRa 模块为例，其接收灵敏度达到惊人的-140dBm，而超强的信噪比可以让 WINEXT 网关和终端工作在噪声门限以下 20dB。

LoRa 的数据传输速率和数据长度是怎样的？

LoRaWAN 协议定义了一系列的数据传输速率，不同的芯片可供选择的速率范围不同，例如 SX1272 支持 0.3-38.4kbps，SX1276 支持 0.018-38.4kbps 的速率范围。目前 WINEXT 能实现 0.3-37.5kbps 的传输速率。使用 LoRa 设备发送或接收的数据长度有限制，理论来说 SX127x 系列芯片有 256 Bytes 的 FIFO，发射或接收 256 Bytes 都行得通。但是，并不是在任何传输速率下 LoRa 模块的负载长度都能为 256 Bytes。在传输速率较低的情况下，一次传输 256 Bytes 需要花费的时间极长（可能需要花费几秒甚至更长），这不利于抗干扰和交互，因此在技术处理上一般建议用户将一条长数据分割成数条小数据来进行传输。

我可以注册一个设备到多个应用程序吗？

可以，但是该应用程序一次只能激活一个应用程序。你所用的密匙决定了应用程序，应用程序必须用密匙去开启。

我可以在我的服务器上运行私有云中间件吗？

可以，如果你了解你所做的，你可以运行我们的一个私有部署。

单通道网关与真正网关的区别是什么？

一个真正的网关能够在同一时间监听多个（至少 8 个）通道和所有扩频因子的信息。单通道网关固定为监听一个通道和扩频因子的信息，因此他们只能接收 2%的信息，除非你特别配置你的节点，使得节点以与你单通道网关相应的配置发送信息。由于 LoRaWAN 是一种扩频无线协议，单通道网关不兼容 LoRaWAN. 他们只是被用来简化 LoRa 通讯或者测试，而不被私有云中间件服务支持。

我的节点提示“发送成功”，但是我的应用程序没有收到任何信息，这是怎么回事？

“发送成功”只是指节点发送了信息，而不是指网关也收到了此信息。如果你在操作台或者应用程序里找不到此信息，那么你的节点也许正好不在网关的有效范围内。

我的节点提示“连接不接受：被拒绝了”。这是为什么呢？

在 LoRaWAN 中，我们只提示“连接被接受”。如果连接不接受，节点将收不到任何提示。因此“拒绝连接”可能是由其他各种原因造成，而不是服务器拒绝连接。 首先，你需要确保你的节点在网关的有效范围内，因为十之八九是因为节点不在网关的有效范围内而造成连接不接受。如果你确定你的节点是在网关有效范围内，你应该查一下设置（AppEUI, DevEUI and AppKey)是否与后台相匹配，同时也查一下他们是否输入了正确的格式(msb, lsb, hex).

我不再从我的 ABP 设备接收数据了，这是怎么回事？

当你 ABP 设备的信息无法到达你的后台，你的设备可能重启了。当这一切发生的时候，你的设备通常帧计数器重置为 0，然后重新开始发送。然而你的后台将会把这些信息标记为可能的重播攻击，从而终止他们。通过禁用帧计数器检查你的设备设置，可以使这种行为被禁用。这将使你的设备容易受到重播攻击，这也就是为什么我们更偏向于推荐 OTAA，而非 ABP。