

低速无线个域网仿真

本章介绍低速无线个域网的基本特点、原理、应用场景等,利用 NS-2 仿真分析代表性的低速无线个域网标准,旨在让读者了解低速无线个域网的设计原理和应用过程。

5.1 预备知识

5.1.1 低速无线个域网

无线个域网(Wireless Personal Area Network, WPAN)的覆盖范围通常在半径 10m 以内,主要技术包括蓝牙(Bluetooth)、超宽带(Ultra Wide Band, UWB)、ZigBee 等。具体可参见《无线网络技术》第 8 章内容。

低速无线个域网(Low Rate WPAN, LR-WPAN)可实现低速率、低功耗、低成本设备之间的无线通信,具有易安装、可靠传输、短距离通信、极低功耗和协议简单灵活等特点,IEEE 802.15.4 是其核心标准。在 IEEE 802.15.4 基础上,ZigBee 提供了完整网络层和应用层协议,而 6LoWPAN 引入 IPv6 使得 LR-WPAN 设备方便接入互联网。LR-WPAN 广泛用于智能家居、远程医疗和工业控制等领域。目前,LR-WPAN 路由尚无统一标准,大多为 AODV 简化或改进版本。

LR-WPAN 核心组件:①**协调器**,控制和监控已建立网络。根据范围可分为 PAN 协调器和普通协调器两种,前者充当个域网(Personal Area Network, PAN)整体的协调器,而后者则在集群(网络的一部分)范围内发挥作用,与 PAN 协调器通信。②**设备/端节点**,可以是精简功能设备(Reduced Function Device, RFD)或全功能设备(Full Function Device, FFD),其中 FFD 支持 49 个原语,可作为 PAN 协调器、普通协调器、端节点(设备),任何非协调器的设备都是端节点(设备);RFD 只能作为端节点(设备),最多支持 38 个原语,适用于电灯开关等简单应用,定期向协调器发送关于其监控设备状态的信息。③**个人操作空间**(Personal Operating Space, POS),节点在所有方向上的操作范围,且无论处于运动状态还是静止状态,都是一个常数。

在网络拓扑方面,LR-WPAN 支持三种不同类型的拓扑结构(见图 5.1),即星状(PAN 协调器主控制)、P2P(FFD 与 PAN 协调器、FFD 间都可通信)和簇树(簇头(FFD)与 PAN 协调器通信)。

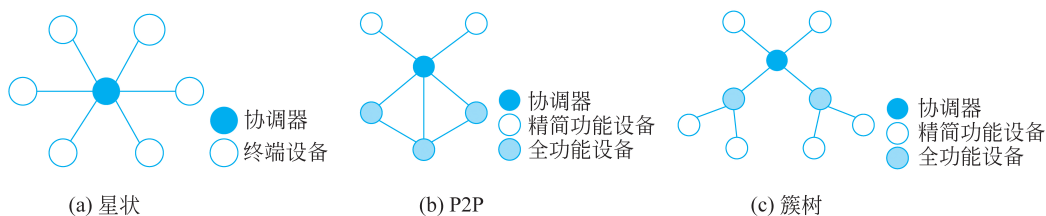


图 5.1 LR-WPAN 支持的拓扑类型

5.1.2 IEEE 802.15.4

IEEE 802.15.4 由物理层和 MAC 层组成,上层为不同应用程序,如图 5.2 所示。物理层负责启用和停止无线电收发、能量检测、链路质量指示、信道选择、空闲信道评估,以及在物理介质上发送和接收数据包。MAC 层则负责信标管理、通道访问、保证时隙管理、帧验证、确认帧交付、关联和解除关联等。

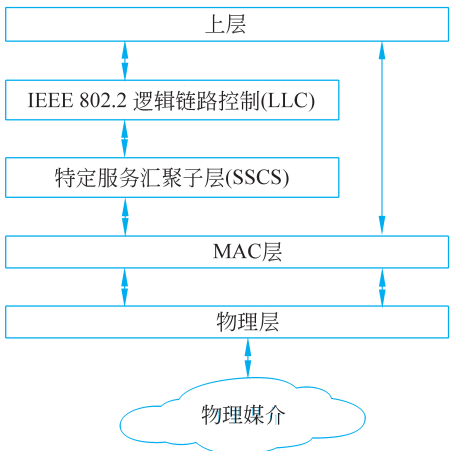


图 5.2 IEEE 802.15.4 架构

为使网络通信能高效、节能,IEEE 802.15.4 定义了如图 5.3 所示的超帧(Superframe)结构,其是一种时间分配机制,可确保网络设备在正确时间进行通信。

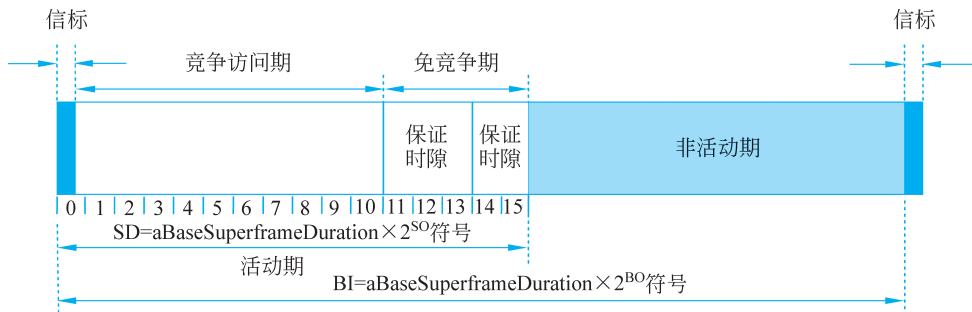


图 5.3 超帧结构示意图

在竞争访问周期内,网络中每个设备采用 CSMA/CA 方式竞争传输数据。在无竞争

周期或保证时隙内,低时延应用的设备被赋予了通道专有权,可直接传输数据,且在竞争访问周期后立即开始。超帧分活跃期和可选非活跃期两部分,整个超帧的信标间隔(Beacon Interval, BI)与超帧规范中的信标阶(Beacon Order, BO)相关,而超帧持续时间(Superframe Duration, SD)则由超帧阶(Superframe Order, SO)决定。在 IEEE 802.15.4—2006 规范中, aBaseSlotDuration 为 60symbols, 而 aBaseSuperframeDuration 为 aBaseSlotDuration 乘以时隙数。若时隙数为 16(见图 5.3),则其 symbol 数量为 960。由于 $SD = aBaseSuperframeDuration \times 2^{SO}$, 且假设 SO 为 0, 则 SD 的 symbol 数为 960。在 2.4GHz 信道中,每个 symbol 时间为 $16\mu s$, 因而 SD 的持续时间为 15.36ms。调整 SO 值可以控制超帧的持续时间,而调整 BO 可控制超帧间隔。

通常 IEEE 802.15.4 网络可工作于信标模式和非信标模式。信标模式下,协调器定期发送数据包或信标,促使所有节点在信标之间进入睡眠态,并在信标定时器到期时醒来,准备接收来自协调器的信标。超帧结构仅适用于信标模式网络。在非信标网络中,超帧结构被禁用,节点通过 CSMA/CA 竞争信道访问。

5.2 实验环境

本实验采用 NS-2 仿真器源码(使用自带的 Zheng 等开发的 802.15.4 模块,具体网址见附录 B 说明和“无线网络技术教学研究平台”),Linux 系统为 Ubuntu 20.04 且运行在 VirtualBox 6.18 中。具体环境安装详见导引实验。

5.3 实验步骤

5.3.1 星状拓扑实验

构建如图 5.4 所示的星状拓扑,包含 7 个节点,其中节点 0 为 PAN 协调器节点,开启 Beacon 功能,节点 1~6 为终端节点。数据在节点 0→1 和节点 0→3 间传输,采用 TCP/FTP 应用。

具体实验代码如下所示。配置协调器和终端节点时需使用不同语句。对于 PAN 协调器节点,执行 `$node SCS StartPANCoord <txBeacon> <BO> <SO>`, txBeacon 设置为 0 或 1,表示关闭或开启信标模式;而对于终端节点则执行 `$node SCS startDevice <isFFD> <assopermit> <txBeacon> <BO> <SO>`, isFFD 设置为 0 或 1,表示节点是否为全功能节点,全功能节点可作为普通协调器。

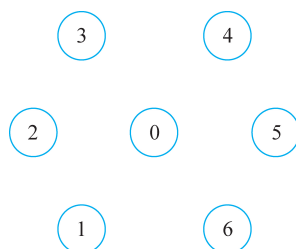


图 5.4 星状实验拓扑

```
set val(chan)          Channel/WirelessChannel
set val(prop)          Propagation/TwoRayGround
set val(netif)         Phy/WirelessPhy/802_15_4
```

```
set val(mac)           Mac/802_15_4
set val(ifq)           Queue/DropTail/PriQueue
set val(ll)            LL
set val(ant)           Antenna/OmniAntenna
set val(ifqlen)        150
set val(nn)            7
set val(rp)            AODV
set val(x)             50
set val(y)             50
set appTime1           7.0
set appTime2           7.1
set appTime3           7.2
set stopTime           100

set ns_                [new Simulator]
set tracefd            [open star.tr w]
$ns_ trace-all $tracefd
set namtrace           [open star.nam w]
$ns_ namtrace-all-wireless $namtrace $val(x) $val(y)
#默认关闭,开启后其他 wpanNam 命令才能工作
Mac/802_15_4 wpanNam namStatus on
Mac/802_15_4 wpanCmd verbose on
#设置 CStresh_和 RXThresh_阈值,传播模型为 TwoRayGround
set dist(5m) 7.69113e-06
set dist(10m) 1.92278e-06
set dist(15m) 8.54570e-07
set dist(20m) 4.80696e-07
set dist(25m) 3.07645e-07
set dist(30m) 2.13643e-07
set dist(35m) 1.56962e-07
set dist(40m) 1.20174e-07
Phy/WirelessPhy set CStresh_ $dist(15m)
Phy/WirelessPhy set RXThresh_ $dist(15m)
#创建拓扑和信道实例
set topo               [new Topography]
$topo load_flatgrid $val(x) $val(y)
set god_               [create-god $val(nn)]
set chan_1_            [new $val(chan)]
#配置节点,务必在实例化节点前完成
$ns_ node-config -adhocRouting $val(rp) \
    -llType $val(ll) \
    -macType $val(mac) \
    -ifqType $val(ifq) \
```

```

        -ifqLen $val(ifqlen) \
        -antType $val(ant) \
        -propType $val(prop) \
        -phyType $val(netif) \
        -topoInstance $topo \
        -agentTrace ON \
        -routerTrace OFF \
        -macTrace ON \
        -movementTrace OFF \
        -channel $chan_1_

#实例化节点, 设置节点位置, 设置节点类型
for {set i 0} {$i < $val(nn)} {incr i} {
    set node_($i) [$ns_node]
    $node_($i) random-motion 0
}
$node_(0) set X_ 25
$node_(0) set Y_ 25
$node_(0) set Z_ 0
$node_(1) set X_ 20
$node_(1) set Y_ 16.34
$node_(1) set Z_ 0
$node_(2) set X_ 15
$node_(2) set Y_ 25
$node_(2) set Z_ 0
$node_(3) set X_ 20
$node_(3) set Y_ 33.66
$node_(3) set Z_ 0
$node_(4) set X_ 30
$node_(4) set Y_ 33.66
$node_(4) set Z_ 0
$node_(5) set X_ 35
$node_(5) set Y_ 25
$node_(5) set Z_ 0
$node_(6) set X_ 30
$node_(6) set Y_ 16.34
$node_(6) set Z_ 0
for {set i 0} {$i < $val(nn)} {incr i} {
    $ns_initial_node_pos $node_($i) 3
}

# PAN 协调器节点标签
$ns_at 0.0 "$node_(0) NodeLabel PAN Coord"

#启动协调器, 默认 beacon 使能, BO 和 SO 都为 3

```

```

$ns_ at 0.0 "$node_(0) sscs startPANCoord"
#启动 FFD 节点, beacon 关闭, BO 和 SO 也为 3
$ns_ at 0.5 "$node_(1) sscs startDevice 1 0"
$ns_ at 1.5 "$node_(2) sscs startDevice 1 0"
$ns_ at 2.5 "$node_(3) sscs startDevice 1 0"
$ns_ at 3.5 "$node_(4) sscs startDevice 1 0"
$ns_ at 4.5 "$node_(5) sscs startDevice 1 0"
$ns_ at 5.5 "$node_(6) sscs startDevice 1 0"
#流建立函数
proc ftptraffic { src dst starttime } {
    global ns_ node_
    set tcp($src) [new Agent/TCP]
    $tcp($src) set packetSize_ 50
    set sink($dst) [new Agent/TCPSink]
    $ns_ attach-agent $node_($src) $tcp($src)
    $ns_ attach-agent $node_($dst) $sink($dst)
    $ns_ connect $tcp($src) $sink($dst)
    set ftp($src) [new Application/FTP]
    $ftp($src) attach-agent $tcp($src)
    $ns_ at $starttime "$ftp($src) start"
}
ftptraffic 0 1 $appTime1
ftptraffic 0 3 $appTime3
#仿真结束处理
for {set i 0} {$i < $val(nn)} {incr i} {
    $ns_ at $stopTime "$node_($i) reset"
}
proc stop {} {
    global ns_ tracefd
    $ns_ flush-trace
    close $tracefd
    close $namtrace
}
$ns_ at $stopTime "stop"
$ns_ at $stopTime "$ns_ halt"
#开始仿真
$ns_ run

```

根据上述脚本代码,可按如下操作完成实验。

(1) 使用 **Ctrl + Alt + T** 组合键打开终端,并用 **cd /home/wiotlab/WIoTlab/exps/lab5** 命令切换路径,接着使用 **touch Star.tcl** 命令创建脚本文件,将上述代码录入该文件中。

(2) 在命令行终端运行 **ns Star.tcl** 执行脚本,成功后会得到 Star.tr、Star.nam 等数据

文件,其中 Star.tr 是待处理的记录。

(3) 分析协议平均吞吐量(4.3.2 节对应代码存为 `tput.awk` 文件)、数据包投递率(4.3.2 节对应代码存为 `pdr.awk` 文件),具体如下。

平均吞吐量: 在命令行终端运行 `awk -f tput.awk Star.tr`,可得到结果 Average Throughput [kbps] = 61.88 startTime=7.24 stopTime=99.99。

数据包投递率: 在命令行终端运行 `awk -f pdr.awk Star.tr`,可得到结果 Sent: 16002, Received:15837, Delivery Ratio:0.9897。

(4) 若需查看 nam 动画,可在命令行终端运行 `nam Star.nam`,查看仿真过程。

5.3.2 P2P 拓扑实验

构建如图 5.5 所示的 P2P 实验拓扑,包含 11 个节点:节点 0 为 PAN 协调器节点,开启了 Beacon 功能;节点 1~5 为普通协调器节点;节点 6~10 为普通终端节点。8.3s 和 8.6s 时节点 1→6 和 4→10 产生 TCP/FTP 流,并在 100s 时结束。

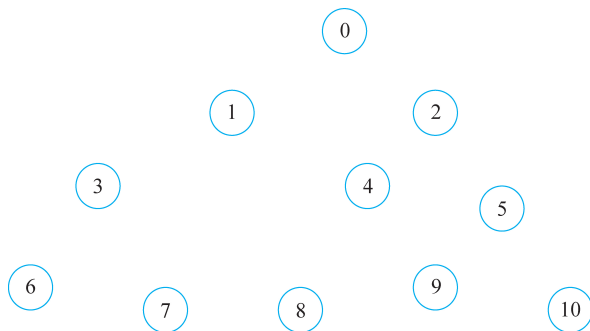


图 5.5 P2P 实验拓扑

具体实验代码如下所示。配置协调器和终端节点时需使用不同语句,如星状拓扑实验所述。`$node sscs startBeacon <BO> <SO>`表示启动节点信标模式,并传入 BO 和 SO 参数。

```
set val(chan) Channel/WirelessChannel
set val(prop) Propagation/TwoRayGround
set val(netif) Phy/WirelessPhy/802_15_4
set val(mac) Mac/802_15_4
set val(ifq) Queue/DropTail/PriQueue
set val(ll) LL
set val(ant) Antenna/OmniAntenna
set val(ifqlen) 50
set val(nn) 11
set val(rp) AODV
set val(x) 50
set val(y) 50
```

```
set appTime1      8.3
set appTime2      8.6
set stopTime      100

Mac/802_15_4 wpanCmd verbose on
Mac/802_15_4 wpanNam namStatus on

# Initialize Global Variables
set ns_           [new Simulator]
set tracefd       [open P2P.tr w]
$ns_ trace-all $tracefd
set namtrace       [open P2P.nam w]
$ns_ namtrace-all-wireless $namtrace $val(x) $val(y)

set dist(15m) 8.54570e-07
Phy/WirelessPhy set CStresh_ $dist(15m)
Phy/WirelessPhy set RXThresh_ $dist(15m)

# set up topography object
set topo          [new Topography]
$topo load_flatgrid $val(x) $val(y)

# Create God
set god_ [create-god $val(nn)]

set chan_1_ [new $val(chan)]

# configure node

$ns_ node-config -adhocRouting $val(rp) \
    -llType $val(ll) \
    -macType $val(mac) \
    -ifqType $val(ifq) \
    -ifqLen $val(ifqlen) \
    -antType $val(ant) \
    -propType $val(prop) \
    -phyType $val(netif) \
    -topoInstance $topo \
    -agentTrace ON \
    -routerTrace OFF \
    -macTrace ON \
    -movementTrace OFF \
    -channel $chan_1_
```

```

for {set i 0} {$i < $val(nn) } {incr i} {
    set node_($i) [$ns_node]
    $node_($i) random-motion 0
}

$node_(0) set X_ 30
$node_(0) set Y_ 40
$node_(0) set Z_ 0.000000000000
$node_(1) set X_ 22
$node_(1) set Y_ 32
$node_(1) set Z_ 0.000000000000
$node_(2) set X_ 35
$node_(2) set Y_ 32
$node_(2) set Z_ 0.000000000000
$node_(3) set X_ 13.5
$node_(3) set Y_ 24
$node_(3) set Z_ 0.000000000000
$node_(4) set X_ 29
$node_(4) set Y_ 24
$node_(4) set Z_ 0.000000000000
$node_(5) set X_ 40
$node_(5) set Y_ 23
$node_(5) set Z_ 0.000000000000
$node_(6) set X_ 9
$node_(6) set Y_ 14
$node_(6) set Z_ 0.000000000000
$node_(7) set X_ 16.5
$node_(7) set Y_ 13
$node_(7) set Z_ 0.000000000000
$node_(8) set X_ 24
$node_(8) set Y_ 13
$node_(8) set Z_ 0.000000000000
$node_(9) set X_ 34
$node_(9) set Y_ 13.5
$node_(9) set Z_ 0.000000000000
$node_(10) set X_ 44
$node_(10) set Y_ 13
$node_(10) set Z_ 0.000000000000

$ns_at 0.0 "$node_(0) NodeLabel PAN Coord"
$ns_at 0.0 "$node_(0) sscs startPANCoord 1"
$ns_at 0.5 "$node_(1) sscs startDevice 1 1 1"
$ns_at 1.5 "$node_(2) sscs startDevice 1 1 1"
$ns_at 2.5 "$node_(3) sscs startDevice 1 1 1"

```

```
$ns_at 3.5 "$node_(4) sscs startDevice 1 1 1"
$ns_at 4.5 "$node_(5) sscs startDevice 1 1 1"
$ns_at 5.5 "$node_(6) sscs startDevice 0"
$ns_at 5.8 "$node_(7) sscs startDevice 0"
$ns_at 6.5 "$node_(8) sscs startDevice 0"
$ns_at 6.8 "$node_(9) sscs startDevice 0"
$ns_at 7.0 "$node_(10) sscs startDevice 0"
$ns_at 6.0 "$node_(3) sscs stopBeacon"
$ns_at 8.0 "$node_(3) sscs startBeacon"
$ns_at 9.0 "$node_(5) sscs startBeacon 4 4"
$ns_at 10.0 "$node_(4) sscs stopBeacon"
for {set i 0} {$i < $val(nn)} {incr i} {
    $ns_initial_node_pos $node_($i) 2
}

proc ftptraffic { src dst starttime } {
    global ns_node_
    set tcp($src) [new Agent/TCP]
    $tcp($src) set packetSize_ 50
    set sink($dst) [new Agent/TCPSink]
    $ns_attach-agent $node_($src) $tcp($src)
    $ns_attach-agent $node_($dst) $sink($dst)
    $ns_connect $tcp($src) $sink($dst)
    set ftp($src) [new Application/FTP]
    $ftp($src) attach-agent $tcp($src)
    $ns_at $starttime "$ftp($src) start"
}

ftptraffic 1 6 $appTime1
ftptraffic 4 10 $appTime2
$ns_at $appTime1 "$node_(1) add-mark m1 blue circle"
# $ns_at $stopTime "$node_(1) delete-mark m1"
$ns_at $appTime1 "$node_(6) add-mark m2 blue circle"
$ns_at $appTime2 "$node_(4) add-mark m3 green4 circle"
$ns_at $appTime2 "$node_(10) add-mark m4 green4 circle"
Mac/802_15_4 wpanNam FlowClr -p AODV -c tomato
Mac/802_15_4 wpanNam FlowClr -p ARP -c green
Mac/802_15_4 wpanNam FlowClr -p MAC -c navy
Mac/802_15_4 wpanNam FlowClr -p tcp -s 1 -d 6 -c blue
Mac/802_15_4 wpanNam FlowClr -p ack -s 6 -d 1 -c blue
Mac/802_15_4 wpanNam FlowClr -p tcp -s 4 -d 10 -c green4
Mac/802_15_4 wpanNam FlowClr -p ack -s 10 -d 4 -c green4

proc stop {} {
```