Luiz Fernando Schrickte

PROJETO, IMPLEMENTAÇÃO E AVALIAÇÃO DE DESEMPENHO DE NÓS E GATEWAY 6LOWPAN

Dissertação submetida ao Programa de Pós-graduação em Engenharia de Automação e Sistemas para a obtenção do Grau de Mestre em Engenharia de Automação e Sistemas.
Orientador
Universidade Federal de Santa Catarina: Prof. Dr. Carlos Barros Montez
Coorientador
Universidade Federal de Santa Catarina: Prof. Dr. Rômulo Silva de Oliveira

Florianópolis

2013

Ficha de identificação da obra elaborada pelo autor através do
Programa de Geração Automática da Biblioteca Universitária da UFSC.

A ficha de identificação é elaborada pelo próprio autor

Maiores informações em:
http://portalbu.ufsc.br/ficha

Luiz Fernando Schrickte

PROJETO, IMPLEMENTAÇÃO E AVALIAÇÃO DE DESEMPENHO DE NÓS E GATEWAY 6LOWPAN

Esta Dissertação foi julgada aprovada para a obtenção do Título de "Mestre em Engenharia de Automação e Sistemas", e aprovada em sua forma final pelo Programa de Pós-graduação em Engenharia de Automação e Sistemas.

Florianópolis, 20 de setembro 2013.

Prof. Dr. Jomi Fred Hübner
Coordenador
Universidade Federal de Santa Catarina

Banca Examinadora:

Prof. Dr. Carlos Barros Montez
Universidade Federal de Santa Catarina

Prof. Dr. Carlos Maziero
Universidade Tecnológica Federal do Paraná

Prof. Dr. Eraldo Silveira e Silva
Instituto Federal de Santa Catarina

Prof. Dr. Max Hering de Queiroz
Universidade Federal de Santa Catarina

Este trabalho é dedicado aos meus pais Elizabete e Walter e à minha irmã Mariliz

AGRADECIMENTOS

Este trabalho contou com imenso apoio da empresa Boreste em termos técnicos e financeiros. Sou muito grato a isso. Ao apoio do meu amigo e sócio Adrián Fritz, fundador da empresa, devo não somente a gratidão por me permitir a realização deste trabalho, mas também pela realização do próprio mestrado. As saídas em horário de trabalho para participar das aulas, as longas conversas a respeito do tema e o seu vasto conhecimento técnico e prático sempre disponíveis foram imprescindíveis para que este projeto pudesse ser realizado.

Agradeço à minha família. Foram essenciais. Elaborar um trabalho com essas proporções sem abrir mão da jornada diária de trabalho exigiu apoio diário. Além disso, em várias ocasiões tive que abrir mão de dar a eles a atenção que merecem. Muito obrigado Elizabete Schrickte, Walter Schrickte e Mariliz Regina Schrickte, eu não seria nada sem vocês. Estendo os agradecimentos à minha namorada Graziane Ferreira Paulo, que talvez tenha sido a mais paciente de todas as pessoas envolvidas ao me acompanhar nos finais de semana em casa em frente ao computador e, é claro, por me apoiar sempre.

Aos demais membros da Boreste também tenho muito a agradecer. Sou grato ao Daniel Kraemer, que participou do trabalho ativamente, inclusive ao elaborar seu projeto de fim de curso em tema relacionado, e sempre ajudou com o que podia prontamente. Ao João Filipe Silva Costa, com seu enorme talento para a eletrônica e sua disponibilidade inclusive aos finais de semana para me ajudar. Ao Amilcar Bodini, grande mestre no desenvolvimento de hardware.

Aos meus grandes amigos também sou muito grato. Ao Diego Machado Vieira, com o qual dividi apartamento durante boa parte do trabalho, agradeço pelas inúmeras vezes em que discutiu, estudou junto comigo e ajudou a resolver os problemas que apareceram. Agradeço também ao Eduardo Hülse, com sua capacidade de se envolver com o assunto e participar e ajudar nos momentos de conquista e também nos momentos de dificuldade. E finalmente ao Felipe Augusto de Souza, pela força e apoio em todas as horas em que precisei. Vocês três são exemplos para mim.

À minha tia, Norma Schrickte, ao meu tio, Sylvio Monteiro Júnior, e à minha prima, Daniela Schrickte Stoll, devo a gratidão por sempre me apoiarem, aconselharem e estarem junto comigo em momentos muito importantes na realização deste trabalho e no começo da minha vida pós graduação. Vocês são minha segunda família.

Aos meus orientadores Prof. Carlos Barros Montez e Prof. Rômulo Silva de Oliveira agradeço pela orientação, correções e sugestões, assim

como pela compreensão com a minha falta de tempo e demora em apresentar resultados em alguns momentos.

Agradeço ao meu amigo e colega de trabalho Helmuth Hass por sua ajuda técnica, apoio e preocupação na realização deste trabalho.

Agradeço também à FINEP por ter apoiado financeiramente a realização deste trabalho.

RESUMO

O conceito de Internet das Coisas vem ganhando força, promovendo a integração dos mais simples dispositivos à rede mundial de computadores. O padrão 6LoWPAN permite que tais dispositivos, mesmo com poucos recursos de energia, processamento e memória, conversem com dispositivos comuns compatíveis com o padrão IPv6 através do uso de técnicas de compressão e fragmentação. O presente trabalho contempla o projeto e implementação de nós que utilizam o padrão 6LoWPAN, assim como um Gateway para integração dos nós com uma rede Ethernet. O desempenho de tais dispositivos é avaliado e suas funcionalidades validadas. O padrão 6LoWPAN é analisado e comparado ao uso do IPv6 sem compressão em testes práticos.

Palavras-chave: Redes de Sensores sem Fio. 6LoWPAN Gateway. IEEE 802.15.4

ABSTRACT

The Internet of Things concept is getting stronger by connecting the simplest devices to the Internet. 6LoWPAN allows these devices, despite their limitations of power availability, processing speed and memory, to communicate with IPv6 regular devices by the use of compression and fragmentation techniques. This work contemplates the project and implementation of 6LoWPAN nodes and a gateway for their integration to an Ethernet network. The performance of the devices is evaluated and their functionality validated. The 6LoWPAN specification is analysed and compared to the use of IPv6 without compression in practical tests.

Keywords: Wireless Sensor Networks. 6LoWPAN Gateway. IEEE 802.15.4

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

SUMÁRIO

1 INTRODUÇÃO

As primeiras redes surgiram na década de 1960, diante da necessidade de transferir informação entre computadores. O uso de cabos foi natural, dado o conhecimento e a disponibilidade deste tipo de meio, prevalecendo como o meio físico mais comum até os dias de hoje, tendo obtido grandes avanços desde o surgimento das primeiras redes. A comunicação via cabo teve entre seus precursores o telégrafo, cuja transmissão de dados podia ser acompanhada facilmente por um olho humano atento. A comunicação evoluiu ao longo das últimas décadas, atingindo hoje velocidades antes inimagináveis. É o caso de redes que usam fibra ótica como meio físico, cuja taxa de transferência chega a Terabits por segundo ao se beneficiar da luz como forma de transmissão dos dados.

A partir da década de 1990 redes de comunicação sem fio passaram a se difundir, principalmente em ambientes domésticos. Tais redes, como diz o próprio nome, dispensam cabos na troca de dados entre dispositivos. As principais vantagens das redes sem fio sobre as redes cabeadas são:

- Possibilidade de uso em equipamentos móveis, como em telefones celulares.
- Simplicidade e baixo custo de instalação, pois há necessidade mínima de infra-estrutura na maioria dos casos. Um ponto de rede sem fio usualmente tem custo muito baixo em relação a um ponto de rede cabeada.
- Flexibilidade para mudanças na organização dos nós da rede.
- Imunidade a descargas elétricas conduzidas e a perturbações na rede de comunicação que possam danificar equipamentos conectados.
- Uso em situações em que cabos não são viáveis, como na comunicação com o robô Curiosity em Marte em recente missão da NASA.

Uma das aplicações mais conhecidas para as redes sem fio é a conexão de computadores à Internet em redes domésticas e empresariais. Apesar de inicialmente ser associada apenas a estes cenários, as redes sem fio são utilizadas em um amplo espectro de aplicações, desde redes de celulares e comunicação entre satélites, até aplicações em equipamentos no chão de fábrica ou mesmo em dispositivos de monitoramento ambiental.

Em um ambiente industrial, as vantagens das redes sem fio podem ser determinantes se traduzidas em facilidade de instalação de novos

equipamentos, mudança de *layout* da fábrica, uso em ambientes hostis, proteção elétrica dos equipamentos, dentre outras oportunidades de melhoria. Tais vantagens, porém, não foram suficientes até agora para garantir a adoção significativa de redes sem fio na indústria. Isto se deve à ausência de confiabilidade e segurança dos dados neste tipo de tecnologia em relação às redes cabeadas, problemas que também vêm sido resolvidos nos últimos anos.

As redes sem fio utilizam um meio "aberto", e, sem medidas de segurança adequadas, é simples para um usuário mal intencionado capturar pacotes, inserir pacotes maliciosos ou mesmo sobrecarregar a rede (WILLIG; MATHEUS; WOLISZ, 2005). Além disso, há outros fatores a serem considerados na adoção deste tipo de tecnologia, conforme citado e analisado em (KOUMPIS et al., 2005): robustez, tolerância a falhas, imunidade à interferência, interoperabilidade, entre outros.

Redes sem fio podem estar presentes em equipamentos de grande porte assim como em dispositivos diminutos. A miniaturização dos semicondutores vem permitindo o uso da eletrônica em dispositivos cada vez mais simples. Simultaneamente, a adoção da Internet está crescendo e permitindo a interconexão entre os mais variados equipamentos. A combinação destes fatos resulta no conceito da Internet das Coisas, onde todos os dispositivos, desde a simples cafeteira na cozinha até o automóvel na garagem, estão conectados à grande rede. Tais dispositivos podem gerar informações e serem comandados diretamente, sem a necessidade de um dispositivo centralizador de maior capacidade. Para tal, devem ser compatíveis com os padrões da Internet.

As limitações de recursos desses dispositivos, porém, impõem alguns obstáculos à compatibilidade com alguns desses padrões. Estas dificuldades são resolvidas de múltiplas formas, dentre elas através da adaptação dos padrões utilizados na Internet para que se tornem adequadas para estes equipamentos. Tais adaptações, porém, devem ser transparentes aos demais componentes da rede, de tal maneira que estes não precisem alterar sua forma de se comunicar.

O padrão 6LoWPAN, acrônimo para *IPv6 over Low Power Wireless Personal Area Networks*, é um padrão recente que estabelece uma camada de adaptação para permitir a adaptação dos padrões da Internet a dispositivos com limitações de recursos. Este padrão é discutido e avaliado ao longo deste trabalho. O 6LoWPAN é uma adaptação do IPv6, o protocolo padrão da camada de rede da Internet que está sendo paulatinamente adotado. O projeto e implementação de nós e *gateway* compatíveis com 6LoWPAN são abordados neste trabalho. O 6LoWPAN é analisado através de experimentos realizados utilizando os dispositivos desenvolvidos.

1.1 OBJETIVOS

1.1.1 Motivação

O conceito da Internet das Coisas expande a abrangência da palavra conectividade como a conhecemos hoje ao permitir a interconexão entre todos os dispositivos eletrônicos, do mais complexo ao mais simples. O conceito abre portas para as mais diversas aplicações, como na automação de sistemas, em equipamentos de monitoramento, ou até sob a forma de ferramentas que potencializam e facilitam o acesso a dados importantes remotamente de forma padrão. Atualmente a conexão entre dispositivos simples em redes privadas é usualmente realizada via protocolos fechados. Tal abordagem impede que estes dispositivos sejam compatíveis com equipamentos de outros fabricantes e mesmo com os protocolos padrões da Internet.

A CISCO estima que em 2020 serão mais de 50 bilhões de dispositivos conectados (PRETZ, 2013). Para que tal feito seja possível vários desafios de engenharia devem ser enfrentados. Dentre os desafios pode-se citar a ausência de padrões definidos, a necessidade de manter a segurança dos dados privados, entre outros. Iniciativas relacionadas ao conceito de Internet das coisas, que visam resolver os desafios apresentados, vêm surgindo. Um exemplo é a criação do website para a Internet das Coisas da IEEE[1], que pretende centralizar os esforços ao redor da implementação deste conceito.

O estágio inicial de adoção do conceito de Internet das Coisas ao mesmo tempo que abre espaço para a exploração de novas tecnologias e possibilidades também implica na escassez de referências e recursos para o seu estudo. Tal fato é visto como uma oportunidade neste trabalho para o desenvolvimento e experimentação com dispositivos e tecnologias que servem de base para a Internet das Coisas.

1.1.2 Objetivo Geral

Integrar uma rede sem fio pessoal (WPAN - do inglês *Wireless Personal Area Network*) de dispositivos de baixo custo a uma rede TCP/IP convencional Ethernet. Os dispositivos na WPAN devem ser capazes de endereçar dispositivos na rede Ethernet assim como devem poder ser endereçados por esta rede cabeada. Tal integração será realizada por um dispositivo *gateway*, a ser desenvolvido neste trabalho, assim como os nós da WPAN.

[1] http://iot.ieee.org

1.1.3 Objetivos Específicos

- Desenvolver dispositivos próprios, modulares, com suporte à comunicação sem fio para formação de uma WPAN utilizando padrões abertos de comunicação compatíveis com a Internet.

- Desenvolver dispositivo ponte (do inglês, *bridge*)que permita a integração da WPAN à rede Ethernet convencional. Dispositivos na rede Ethernet devem acessar os dispositivos na WPAN utilizando TCP/IP padrão, sem alterações.

- Validar e obter indicadores de desempenho da comunicação na rede sem fio formada pelos dispositivos construídos.

- Validar e obter indicadores de desempenho da comunicação na rede formada pelos dispositivos da WPAN integrada à rede Ethernet cabeada via *gateway*.

- Analisar o impacto do uso de possíveis camadas de adaptação entre protocolos para dispositivos com baixa disponibilidade de recursos

1.2 METODOLOGIA

A Metodologia a ser adotada neste trabalho consiste em:

1. Definir requisitos do projeto para integração de dispositivos à Internet das Coisas

2. Especificar, projetar, fabricar e montar dispositivos com comunicação sem fio que atendam os requisitos do projeto definidos.

3. Implementar aplicativos simples de rede, testar e validar a comunicação entre dispositivos na rede WPAN.

4. Especificar forma de construção do *gateway* entre dispositivos e rede Ethernet. Se necessário, realizar projeto de hardware para tal.

5. Implementar *gateway*.

6. Testar, validar e avaliar desempenho da comunicação entre dispositivos nas redes formadas.

1.3 ORGANIZAÇÃO DO TRABALHO

O capítulo 2 explica de forma simples a especificação IEEE 802.15.4 para a comunicação sem fio utilizada no trabalho. A forma como a rede é organizada, os componentes da rede, a arquitetura para transporte de dados e as características da especificação são exploradas. No capítulo 3 o padrão IPv6 e as modificações em relação ao IPv4 são expostos. A camada de adaptação 6LoWPAN é descrita e as suas principais características, pontos-chave do trabalho, são mostradas, como a compressão de cabeçalho, o suporte à fragmentação na camada de enlace e o suporte à autoconfiguração de endereços.

O capítulo 4 oferece uma descrição de trabalhos na área da Internet das Coisas, de análise da camada 6LoWPAN, da implementação de nós compatíveis e *gateways*. No capítulo 5 é exibido o projeto e a implementação da rede sem fio, seus nós e a comunicação entre eles, em termos de hardware e software. A implementação do *gateway* também é abordada neste capítulo, proporcionando inicialmente uma visão geral do dispositivo e do seu papel na rede e em seguida descrevendo as soluções de hardware e software utilizadas. No decorrer da descrição da implementação são apresentadas as ferramentas utilizadas ao longo do trabalho e que tornaram possível sua realização, permitindo que os esforços fossem mais assertivos em direção aos objetivos principais de obter uma rede sem fio, um *gateway* e validar ambos.

No capítulo 6 os resultados de testes com a rede e o *gateway* são discutidos e uma avaliação do desempenho do sistema e do uso do 6LoWPAN é conduzida utilizando dados dos experimentos realizados. As conclusões e perspectivas de trabalhos futuros são apresentadas no capítulo 7.

2 IEEE 802.15.4

2.1 INTRODUÇÃO

A especificação IEEE 802.15.4 define as camadas mais baixas de uma rede pessoal sem fio de baixa velocidade. É considerada a especificação padrão para esse tipo de rede, sendo a mais adotada do seu escopo. Exemplos de protocolos de comunicação que utilizam a IEEE 802.15.4 incluem o Zigbee® (ZIGBEE ALLIANCE, 2013), WirelessHART® (HART COMMUNICATION FOUNDATION, 2013) e 6LoWPAN. A especificação contempla comunicação com velocidades de até 250 kbps, possibilidade de topologia estrela ou *peer-to-peer* e baixo consumo de energia. Uma das principais características para sistemas que possuem requisitos de tempo real é a possibilidade de uso dos *slots* de tempo garantido (GTS, do inglês *Guaranteed Time Slots*), que consistem em períodos de tempo garantidos a determinados dispositivos para se comunicarem. A seguir serão apresentados os principais tópicos a respeito da especificação, baseados no documento oficial do IEEE (COMPUTER SOCIETY, 2006).

2.2 TIPOS DE DISPOSITIVO

Dois tipos diferentes de dispositivos podem participar de uma rede IEEE 802.15.4 – Dispositivos de Função Completa (do inglês *Full Function Device* – FFD) e Dispositivos de Função Reduzida (do inglês *Reduced Function Device* – RFD). Os FFDs podem operar de três modos em uma rede pessoal sem fio: coordenador da rede, coordenador local ou dispositivo regular. RFDs cumprem o papel de dispositivos regulares por natureza. FFDs podem se comunicar entre si ou com RFDs, enquanto RFDs podem se comunicar apenas com FFDs. RFDs são dispositivos simples, como sensores ou atuadores, usualmente com pouca capacidade de memória e processamento.

2.3 TOPOLOGIAS

Existem duas topologias possíveis especificadas na norma IEEE 802.15.4: estrela ou *peer-to-peer*. Na topologia tipo estrela há um dispositivo de controle central, o coordenador da rede pessoal (coordenador da PAN). O

coordenador é responsável por iniciar, terminar e rotear a comunicação na rede. Já na topologia *peer-to-peer* um dispositivo pode se comunicar com outros desde que estejam dentro do alcance do sinal. A topologia *peer-to-peer* permite a implementação de redes *mesh*. As duas topologias podem ser visualizadas na Figura 1.

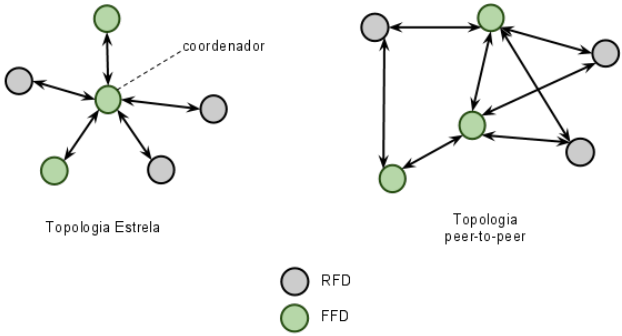


Figura 1 – Topologias na especificação IEEE 802.15.4
Fonte: elaborada pelo autor

2.4 ARQUITETURA PARA TRANSPORTE DE DADOS

A IEEE 802.15.4 especifica a camada física do modelo OSI, assim como na camada de enlace especifica a subcamada de controle de acesso ao meio (MAC). A camada física é responsável pelo transmissor-receptor de rádio frequência e o seu controle de baixo nível. Já o controle de acesso ao meio provê acesso ao meio físico em todas as transferências. É recomendada pela especificação a utilização do padrão IEEE 802.2 para a subcamada de controle de *link* lógico (do inglês *Link Layer Control* – LLC) da camada de enlace, porém esta subcamada não é definida na especificação. A relação entre as camadas pode ser visualizada na Figura 2.

A camada física oferece os seguintes serviços:

- Ativação/desativação do rádio
- Detecção da força do sinal
- Indicação da qualidade do enlace

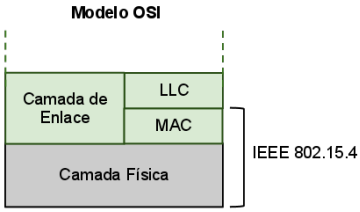


Figura 2 – Relação da especificação com o modelo OSI
Fonte: elaborada pelo autor

- Verificação de canal livre
- Seleção de canais
- Transmissão/recepção de dados através do meio físico

A subcamada de controle de acesso ao meio oferece os seguintes serviços:

- Gerenciamento de *beacons*
- Acesso a canais
- Gerenciamento de *slots* de tempo garantido (GTS)
- Validação de quadros
- Entrega dos quadros de confirmação (ACK)
- Associação e desassociação

2.5 PRINCIPAIS CARACTERÍSTICAS DA ESPECIFICAÇÃO

2.5.1 Modos de Operação

No modo com *beacon* o coordenador da rede estabelece um intervalo delimitado por dois *beacons* e dividido em 16 espaços de tempo iguais. Tal intervalo é denominado *superframe*. O uso de um *superframe* é opcional e o seu formato específico é definido pelo coordenador da rede. Ele pode ter, opcionalmente, uma porção ativa e outra inativa. Na porção inativa

do *superframe*, o coordenador pode entrar em modo de baixo consumo, proporcionando maior economia da bateria.

O *beacon* permite a identificação da rede, o sincronismo entre os participantes e a publicação para os nós da definição da estrutura do *superframe*. Em casos de aplicações de baixa latência ou quando um dispositivo tem que ter uma largura de banda garantida, caso dos sistemas de tempo real, o coordenador pode dedicar partes do *superframe* para dispositivos específicos. Tais porções dedicadas são denominadas *slots* de tempo garantido (GTS). O coordenador da rede pode definir até sete GTS, os quais formam um período livre de contenção (do inglês *Contention Free Period* – CFP). Sempre há uma porção do período ativo, porém, dedicada ao período com contenção (do inglês *Contention Access Period* – CAP), onde dispositivos sem intervalo de tempo garantido transmitem/recebem dados via CSMA/CA e novos dispositivos podem entrar na rede.

Dispositivos que desejam enviar dados entre dois *beacons* competem com outros dispositivos através do mecanismo *slotted* CSMA/CA. Caso o coordenador da rede pessoal não queira utilizar a estrutura de *superframe*, os *beacons* não devem ser enviados, utilizando o modo sem *beacon*. Neste caso a disputa do meio é realizada utilizando CSMA/CA. Neste trabalho o modo sem *beacon* é utilizado para transferência de dados.

2.5.2 Modelo de Transferência de dados

Existem três sentidos de transações de dados em uma rede IEEE 802.15.4 em se tratando de dispositivos:

- Dispositivo $\rightarrow$ Coordenador
- Coordenador $\rightarrow$ Dispositivo
- Dispositivo $\rightarrow$ Dispositivo

Em uma rede com topologia estrela apenas os dois primeiros tipos de transações ocorrem. Os mecanismos de transmissão em uma PAN dependem do suporte a *beacons* na rede. O uso de *beacons* é mandatório caso haja necessidade de sincronização entre dispositivos e/ou os dispositivos da rede sejam de baixa latência. Caso nenhuma dessas características seja necessária na rede, o coordenador pode optar por não usar *beacons* em transferências normais. O uso de *beacons* continua necessário para a descoberta da rede, ou seja, novos nós na rede se utilizam dos *beacons* enviados pelo coordenador para a obtenção de informações da rede e para possíveis pedidos para fazerem parte da mesma.

2.5.2.1 Transferência no sentido Dispositivo para Coordenador

Em uma rede com *beacons* ativados, quando um dispositivo deseja enviar dados para o coordenador, ele deve escutar o meio, esperando por um *beacon*. Quando um *beacon* é encontrado, o dispositivo se sincroniza com a estrutura do *superframe*. No CAP o dispositivo utiliza *slotted* CSMA/CA para transmitir dados para o coordenador. Um quadro de confirmação (ACK) é opcionalmente enviado pelo coordenador. Em uma rede sem *beacons*, o dispositivo simplesmente utiliza *unslotted* CSMA/CA para enviar os dados.

2.5.2.2 Transferência no sentido Coordenador para Dispositivo

Em uma rede com *beacons* ativados, o coordenador informa no *beacon* que há dados pendentes para determinado dispositivo. Tal dispositivo, ao detectar pendência de dados, envia um comando MAC requisitando os dados, utilizando *slotted* CSMA-CA. O coordenador envia então um pacote de confirmação de recepção (ACK), seguido dos dados. Ao final da recepção dos dados o dispositivo envia outra confirmação, opcional, encerrando a transação. É utilizado *unslotted* CSMA-CA quando não há *beacons* ativados.

2.5.2.3 Transferência no sentido Dispositivo para Dispositivo

Este tipo de transferência só é possível em redes *peer-to-peer*. Pode ocorrer entre um dispositivo e qualquer outro que estiver em sua esfera de alcance. Há duas formas deste tipo de transferência ser realizada:

- Cada dispositivo que deseja se comunicar deverá permanecer ouvindo o meio constantemente. Neste caso basta que os dispositivos utilizem *unslotted* CSMA-CA para transmissão.
- Os dispositivos que deseja se comunicar devem estar sincronizados de alguma forma. Tal sincronia, neste caso, encontra-se fora do escopo da especificação e deve ser abordada caso a caso.

2.5.3 Estrutura de Quadros

De acordo com a especificação, as estruturas de quadros foram definidas para manter a complexidade em nível mínimo enquanto promovem robustez suficiente para transmissão em um canal ruidoso. A especificação

define quatro tipos de quadros:

- ***Beacon***: utilizado pelo coordenador de uma rede para transmitir *beacons* e delimitar a estrutura do *superframe*.
- **Data**: quadro usado para o envio de dados.
- **Acknowledgement**: utilizado para confirmação de recepção de quadro de dados.
- **MAC *command***: utilizado no controle da comunicação a nível da subcamada MAC entre dois dispositivos.

2.5.4 Mecanismos para aumentar a probabilidade de sucesso na entrega

A especificação implementa os seguintes mecanismos para aumentar a probabilidade de sucesso na entrega de pacotes:

- Mecanismo CSMA/CA de acesso ao meio: dois mecanismos de acesso ao meio são utilizados na IEEE 802.15.4: *unslotted* e *slotted* CSMA/CA. No caso do *unslotted* CSMA/CA o dispositivo que deseja enviar dados espera por um período aleatório, após o qual escuta o meio. Caso o meio esteja livre, tal dispositivo transmite dados, caso contrário, volta a aguardar um período aleatório antes de nova tentativa de envio de dados. No caso do *slotted* CSMA/CA, o período em que o dispositivo aguarda para tentar novamente transmitir os dados está alinhado com a transmissão de *beacons*.
- Confirmação de quadros: o uso de quadros para confirmar a correta recepção de dados visa melhorar o desempenho da comunicação. Caso a confirmação de quadros esteja ativada e o quadro de confirmação não seja recebido pelo transmissor de dados este pode escolher por retransmitir ou cancelar a transmissão após uma quantidade estabelecida de tentativas.
- Verificação de dados: o uso de CRC de 16 bits permite a detecção de erros em bits na troca de dados.

2.5.5 Baixo consumo de energia

Grande parte dos dispositivos que utilizam a especificação IEEE 802.15.4 tem como única fonte de alimentação baterias. Isto torna

essencial que o consumo de energia seja minimizado, sem comprometer as funcionalidades do dispositivo. Assuntos relacionados à energia estão além do escopo da especificação, ainda assim ela foi desenvolvida considerando baixa disponibilidade de energia.

Apesar de ter como principais alvos dispositivos alimentados a bateria, alguns equipamentos que não possuem restrições de disponibilidade de energia também podem estar na rede e utilizar a especificação. A principal diferença é que equipamentos com limitações de energia podem permanecer boa parte do tempo em estado de baixo consumo, enquanto os demais dispositivos permanecem escutando o canal continuamente. O compromisso entre o consumo de energia e o tempo que o dispositivo permanece ativo na rede é característico de cada aplicação. A IEEE 802.15.4 oferece flexibilidade para tal decisão do projetista.

2.5.6 Segurança

Redes sem fio *ad hoc* possuem alguns fatores que tornam assuntos relacionados à segurança importantes e complexos:

- O ambiente físico em redes sem fio é por natureza compartilhado, facilitando a interceptação de dados e os ataques à rede.
- Dispositivos em LR-WPANs usualmente são de baixo custo e possuem baixo poder computacional, pouca memória e limitações de uso de energia, dificultando, ou tornando impossível, a implementação de algoritmos elaborados de criptografia.
- Não há infra-estrutura fixa de comunicação e dispositivos podem se comunicar brevemente mesmo nunca tendo se comunicado antes.

Muitos dos problemas relacionados à segurança são tratados em camadas superiores, que não pertencem ao escopo desta especificação. Chaves simétricas são utilizadas na criptografia contemplada na especificação, utilizando chaves fornecidas por camadas superiores. O gerenciamento de tais chaves fica restrito a tais camadas. O mecanismo de criptografia fornece os seguintes serviços, em combinações específicas para cada aplicação:

- Confidencialidade: garantia de que dados transferidos são conhecidos apenas pelas partes envolvidas na comunicação.
- Autenticidade: garantia da identidade do dispositivo que enviou os dados, e, por consequência, de que os dados não foram alterados antes de atingir o receptor.

- Proteção contra ataques de *replay*: garante que informações repetidas sejam detectadas.

A proteção de quadros pode ser adaptada quadro a quadro, ou seja, quadros mais importantes podem estar mais protegidos, enquanto quadros cujo conteúdo não apresenta requisitos de segurança fortes podem estar menos protegidos, evitando assim *overhead*.

2.5.7 *Slots* de Tempo Garantido (GTS)

Os GTS são partes do *superframe* alocadas pelo coordenador para comunicação exclusiva com dispositivos visando oferecer um maior grau de determinismo na troca de dados. São úteis principamente em aplicações com requisitos temporais.

A especificação define que no máximo sete GTS podem ser alocados em cada *superframe*. Os dispositivos da rede cujos GTS estão alocados, entretanto, podem variar no decorrer do tempo. A alocação dos GTS é realizada pelo coordenador da rede mediante requisição por parte do dispositivo. Durante um GTS específico, apenas o dispositivo alocado pode se comunicar com o coordenador da rede. O sentido da comunicação é definido para cada *superframe*. A desalocação pode ser solicitada pelo dispositivo ou pode ser realizada pelo coordenador da rede, o qual deve fornecer aos nós informações atualizadas sobre a alocação dos GTS. A troca de informações a respeito da alocação dos GTS é realizada no CAP.

O fato de o período entre *superframes* ser conhecido e de haver um intervalo de tempo dedicado a apenas um dispositivo neste período permite a utilização da especificação IEEE 802.15.4 em aplicações com requisitos de tempo real.

2.6 CONSIDERAÇÕES FINAIS

O objetivo da especificação IEEE 802.15.4 de permitir que dispositivos com baixa disponibilidade de recursos energéticos se comuniquem em redes sem fio é atingido, permitindo que várias especificações de camadas superiores possam se basear neste padrão. As características e funcionalidades oferecidas permitem o cumprimento do objetivo determinado e por consequência estão alinhados com o conceito de Internet das Coisas, tornando a especificação adequada para ser adotada nas camadas inferiores dos dispositivos que se encaixam neste conceito.

3 IPV6 E 6LOWPAN

3.1 INTRODUÇÃO

O IPv6 é o protocolo padrão da camada de rede na Internet. O IPv6 sucede o IPv4, visando estender o limite de endereços definido nessa especificação. As melhorias proporcionadas pelo IPv6, porém, não se restringem à capacidade de endereçamento. Uma gama de modificações foi promovida visando tornar a comunicação mais eficiente entre os dispositivos que utilizam este protocolo, além de adicionar funcionalidades e facilidades à implementação anterior.

A adoção do IPv6 é certa nos próximos anos. Seu uso em redes de dispositivos simples, porém, exige alterações devido a diversas restrições nesses dispositivos. O padrão 6LoWPAN foi proposto exatamente com o objetivo de superar essas restrições, e também será visto neste capítulo.

3.2 ALTERAÇÕES NO IPV6 EM RELAÇÃO AO IPV4

As alterações primárias da especificação IPv6 em relação à especificação IPv4 são, de acordo com a RFC2460 (DEERING; HINDEN, 1998):

1. **Expansão na quantidade de endereços**: o endereço IP na especificação IPv6 possui 128 bits, contra 32 bits da especificação anterior. Além do maior número de nós endereçáveis, isto permite o suporte a mais níveis de hierarquia e opções de auto-configuração mais simples.

2. **Simplificação do cabeçalho**: campos do cabeçalho utilizado nos datagramas IPv4 foram removidos ou tornados opcionais, diminuindo o custo de processamento de pacotes mais transmitidos e o custo da banda necessária para enviar tais dados.

3. **Suporte melhorado para extensões e opções**: o novo cabeçalho permite o encaminhamento mais eficiente de pacotes, limites menos restritos no tamanho das opções e flexibilidade para adição de opções no futuro.

4. **Suporte à marcação de fluxos**: fluxos específicos de pacotes podem receber tratamento especial no caminho até seus destinos.

5. **Recursos de autenticação e privacidade**: extensões de autenticação, integridade de dados e confidencialidade são especificados no IPv6.

Além das mudanças primárias citadas acima, três outras funcionalidades importantes foram implementadas:

3.2.1 SLAAC – Stateless Address Auto-Configuration

A especificação RFC4862 (THOMSON; NARTEN; JINMEI, 2007) define a auto-configuração de endereços para redes IPv6. Dispositivos conectados a uma rede IPv6 podem configurar automaticamente seus próprios endereços. A auto-configuração se utiliza do *Neighbor Discovery Protocol* – NDP, definido na RFC4861 (NARTEN et al., 2007), o qual permite:

- A descoberta de outros dispositivos no mesmo enlace e seus endereços.
- A detecção de endereços duplicados na rede.
- A descoberta de roteadores e servidores DNS.
- A descoberta de prefixos de endereços.
- A manutenção de informações de alcance dos vizinhos da rede.

O sistema não precisa ter IP configurado antes de ser inicializado. Após a inicialização, um endereço preliminar do dispositivo pode ser, na forma mais simples, estabelecido a partir do próprio endereço MAC, baseado no EUI-64, ou formado por esse endereço combinado ao prefixo de rede obtido via NDP de um roteador disponível. Antes de associar este endereço preliminar a uma interface, porém, o nó deve verificar se tal endereço não está sendo utilizado por outro nó no mesmo enlace. Uma solicitação via NDP é enviada à rede e, caso algum nó já possua tal endereço, ele responde. Caso um nó não consiga validar seu endereço preliminar como único na rede, a configuração do endereço deste nó deve ser manual. Caso seja único, a interface de rede assume tal endereço. O trabalho (SHELBY; CHAKRABARTI; NORDMARK, 2012) versa sobre otimizações no protocolo NDP em redes IEEE 802.15.4.

3.2.2 *Multicast*

Multicasting se refere à possibilidade de um mesmo pacote ser enviado para mais de um destinatário na rede em uma única operação de envio.

Opcional na especificação IPv4, tal funcionalidade está implementada na especificação base do IPv6.

3.2.3 Fragmentação de pacotes nas bordas da rede

Roteadores não realizam fragmentação de pacotes em redes IPv6, permitindo maior eficiência no encaminhamento de pacotes na rede. Há três formas de enviar dados maiores que o tamanho máximo:

- Utilizar recursos para determinar o tamanho do MTU no caminho a ser enviado o pacote e adequá-lo a tal valor.
- Fragmentar o pacote nas bordas da rede.
- Não enviar pacotes maiores que o mínimo padrão de 1280 bytes.

3.3 6LOWPAN

LLNs – do inglês *Low Power and Lossy Networks* – são redes com restrições de energia, poder de processamento e memória, cujas conexões entre os nós são caracterizadas por alta perda de pacotes, instabilidade e baixas velocidades. Redes baseadas em IEEE 802.15.4 são classificadas como LLNs. O conceito de Internet das Coisas defende a inclusão do maior ao menor dispositivo à Internet. As restrições de redes IEEE 802.15.4, porém, dificultam a implementação direta do padrão IPv6 da Internet pois:

- O mínimo MTU do IPv6, de 1280 bytes, é muito maior que o tamanho do datagrama da camada MAC definida na IEEE 802.15.4, de 127 bytes. No pior caso, quadros em redes IEEE 802.15.4 deixam apenas 81 bytes para camadas superiores (MONTENEGRO et al., 2007).
- O excesso de dados nos cabeçalhos do IPv6 sobrecarregariam redes IEEE 802.15.4.
- O endereçamento em redes IEEE 802.15.4 de 64 bits, ou 16 bits quando utilizando endereços curtos, é distinto dos endereços de 128 bits do IPv6.
- A quantidade de cálculos mais intensiva demandada pelos mecanismos IPv6 não é adequada para dispositivos com poucos recursos de processamento e energia.

- O IPv6 não define mecanismos de roteamento em redes sem fio com topologia *mesh*.

O uso de protocolos proprietários em LLNs é comum, porém impede a integração direta de tais dispositivos a redes IP e, por consequência, à Internet. O documento RFC4919 (KUSHALNAGAR; MONTENEGRO; SCHUMACHER, 2007) organiza estes problemas e define objetivos para o estabelecimento do IPv6 sobre redes pessoais sem fio de baixo consumo (LoWPANs), procurando atender às peculiaridades deste tipo de rede.

A ideia do 6LoWPAN é estabelecer uma camada de adaptação entre a camada de enlace e de rede do modelo OSI que altere a implementação IPv6 neste tipo de rede para tornar possível seu uso em LLNs. A principal vantagem frente aos demais protocolos que executam sobre a especificação IEEE 802.15.4 se encontra no fato de ser uma rede compatível com IPv6 permitindo a integração direta destes dispositivos a outras redes IP e à Internet. Além disto, a intenção é ser um padrão aberto, universal, em oposição aos protocolos proprietários existentes.

A camada de adaptação do 6LoWPAN, especificada na RFC4944 (MONTENEGRO et al., 2007), prevê as seguintes adequações:

- **Compressão de cabeçalhos IPv6** – o tamanho padrão dos cabeçalhos de um pacote IPv6 é de 40 bytes. Utilizando-se técnicas de compressão eles podem ser reduzidos para até 2 bytes, removendo-se informações não necessárias e derivando-se informações presentes em camadas inferiores.
- **Compressão de cabeçalhos de camadas superiores**, especificamente dos protocolos TCP, UDP e ICMP.
- **Fragmentação dos datagramas IPv6**, para envio em quadros menores da subcamada MAC IEEE 802.15.4.
- **Adição de cabeçalhos e informações** nos pacotes para otimizar as configurações típicas de redes IEEE 802.15.4 com topologias *mesh* e estrela.

As adaptações promovidas na camada 6LoWPAN são explicadas em mais detalhes nas seções a seguir.

3.3.1 Compressão de cabeçalho

Cabeçalhos são estruturas de dados que contêm informações do protocolo sendo utilizado na troca de informações. Apesar de necessários,

usualmente não são úteis para a aplicação, a qual constitui o fim de qualquer dispositivo desenvolvido. Em dispositivos com velocidade limitada e baixa disponibilidade de energia o excesso de informações em cabeçalhos pode prejudicar a comunicação. A compressão de cabeçalhos visa reduzir esse problema, utilizando técnicas que permitem a redução da quantidade de dados de informações do protocolo trocadas sem perdas para a aplicação final.

Na RFC4944 (MONTENEGRO et al., 2007) são definidas as compressões de cabeçalho LOWPAN_HC1 e LOWPAN_HC2. Estas técnicas removem e comprimem informações dos cabeçalhos que são fixas em uma rede 6LoWPAN ou que podem ser obtidas dos cabeçalhos de outras camadas, principalmente da subcamada MAC. Endereços também podem ser comprimidos caso ambos os dispositivos envolvidos na comunicação compartilhem contexto e tenham conhecimento, por exemplo, de um mesmo prefixo utilizado em uma rede. A HC1 comprime cabeçalhos da camada de rede IPv6, enquanto a HC2 permite a compressão de cabeçalhos UDP na camada de transporte. Um cabeçalho IPv6, de 40 bytes, pode ser comprimido a até 2 bytes em determinadas condições.

Na compressão de cabeçalhos 6LoWPAN o tipo de compressão é identificado por um byte denominado *Dispatch* Byte, seguido pelo cabeçalho comprimido. No caso de um cabeçalho sem compressão, há um *Dispatch* Byte específico para IPv6 sem compressão.

Na compressão HC1:

```
+-----------------+---------------+----------+
|   HC1 Dispatch  |   HC1 Header  | Payload  |
+-----------------+---------------+----------+
```

No caso em que não há compressão:

```
+-----------------+---------------+----------+
|  IPv6 Dispatch  |  IPv6 Header  | Payload  |
+-----------------+---------------+----------+
```

Os algoritmos de compressão da RFC4944 foram tornados obsoletos pela RFC6282 (HUI; THUBERT, 2011), que atualiza a RFC4944. A RFC6282 especifica as compressões denominadas LOWPAN_IPHC e LOWPAN_NHC e afirma que a LOWPAN_HC1 e LOWPAN_HC2 não são adequadas para a maioria das redes 6LoWPAN. Pesa contra as formas de compressão definidas na RFC4944 o fato de necessitarem o carregamento do endereço IP parcial ou completo no cabeçalho em casos de comunicação com dispositivos fora do *link* local, cenário que abrange os principais casos de uso do 6LoWPAN.

Em outras palavras, a compressão do cabeçalho para até 2 bytes

utilizando a compressão HC1 é considerada um bom resultado, porém, para atingir tal compressão, ambas as partes na comunicação devem derivar o endereço de informações da camada MAC IEEE 802.15.4. Isto é feito estabelecendo endereços de enlace local de forma que ambos os lados da comunicação tenham previamente estabelecidos os critérios de formação dos endereços (prefixo + identificador da interface), limitando a compressão máxima ao escopo de redes locais. Um endereço global, porém, é necessário para o roteamento e comunicação com redes externas à WPAN. Nestes casos a compressão utilizando IPHC/NHC resulta em pacotes menores em relação à compressão HC1/HC2.

Além disso, a compressão HC2 só permite a compressão de cabeçalhos acima da camada de rede (*Next Headers*) quando estes forem TCP, UDP ou ICMPv6, sem suporte a outros protocolos. A atualização proposta na RFC6282 busca resolver estes problemas.

A dimensão do ganho ao se usar compressão IPHC/NHC em relação às compressões definidas na RFC4944 pode ser visualizada nas figuras 3 e 4. A Figura 3 exibe os resultados após a compressão utilizando HC1. A Figura 4 mostra o resultado ao comprimir os mesmos dados utilizando IPHC.

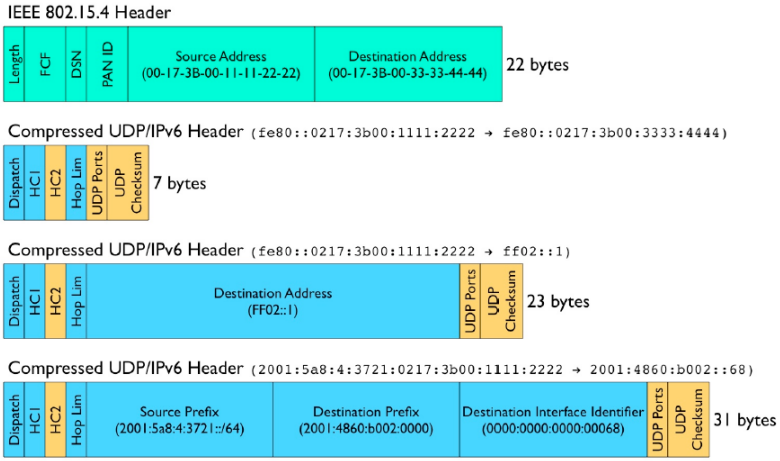


Figura 3 – Compressão LOWPAN_HC1
Fonte: (HUI; CULLER; CHAKRABARTI, 2009)

A primeira barra exibe o cabeçalho padrão MAC da especificação IEEE 802.15.4, que é igual em ambos os casos, uma vez que a camada de adaptação prevista no 6LoWPAN interfere apenas nos cabeçalhos das

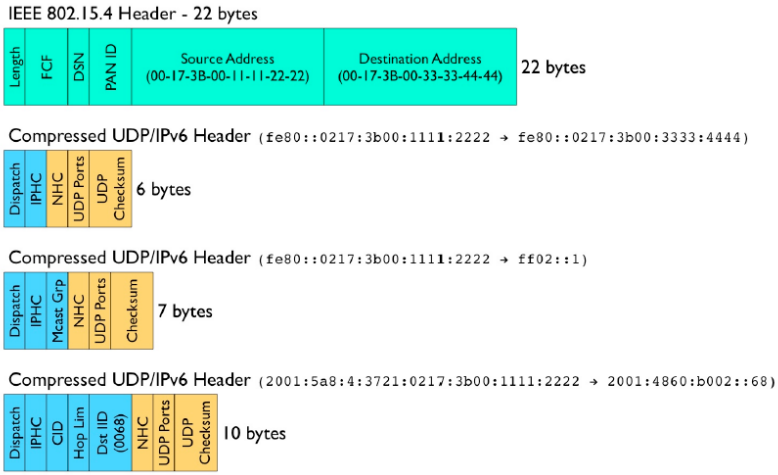


Figura 4 – Compressão LOWPAN_IPHC
Fonte: (HUI; CULLER; CHAKRABARTI, 2009)

camadas de rede e transporte.

Na segunda barra têm-se os cabeçalhos da camada de rede e de transporte considerando-se um pacote enviado no *link* local, *unicast*. Pode-se notar comparando ambas as figuras que a diferença de tamanho é de apenas um byte, com vantagem para a compressão IPHC. A omissão do byte de limite de saltos, (*Hop Limit*, em inglês), neste tipo de compressão justifica a diferença: na HC1/HC2 o valor do limite de saltos deve estar completamente contido no cabeçalho, enquanto na IPHC/NHC quando um dos valores mais comuns é utilizado tal valor por ser omitido e bits no byte de *Dispatch* indicam qual o valor do limite de saltos. O parâmetro de limite de saltos especifica a quantidade máxima de nós no caminho de um pacote antes que o mesmo seja descartado.

Na terceira barra considera-se transmissão *multicast* no *link* local. Neste caso já é notável a melhoria no uso da compressão IPHC/NHC, a qual reduziu os cabeçalhos IP e UDP somados para 7 bytes. Utilizando a compressão HC1/HC2 são obtidos 23 bytes. Da diferença, 1 byte corresponde ao limite de saltos como no caso anterior e os outros 16 bytes correspondem ao endereço de destino que está presente no uso da compressão HC1/HC2 e é omitido no uso da IPHC/NHC. Isto porque a IPHC/NHC se beneficia do uso de endereços de *multicast* conhecidos por ambos os lados, limitando a quantidade de identificadores de grupos *multicast* e permitindo que apenas

um byte do endereço (neste caso específico) sejam carregados no cabeçalho.

Por fim, na última comparação, o pacote é enviado para um nó em uma rede externa 6LoWPAN. Neste caso obteve-se 31 bytes utilizando HC1/HC2 e 10 bytes ao utilizar IPHC/NHC. Enquanto na compressão HC1/HC2 o prefixo da fonte, do destino e o endereço do nó destino estão presentes no cabeçalho, a compressão IPHC/NHC apresenta apenas os dois últimos bytes do identificador de interface de destino, os demais obtidos a partir do uso de técnica de compartilhamento de contexto. No caso da comunicação com nó em rede externa IPv6 tal compartilhamento de contexto não é utilizado e pelo menos o endereço de destino do pacote completo está presente após a compressão do cabeçalho na origem.

Neste trabalho foram utilizadas as formas de compressão definidas na RFC6282. A Figura 5 exibe o cabeçalho padrão IPv6 e as alterações que podem ser promovidas ao se utilizar a compressão IPHC na camada de adaptação 6LoWPAN. Apesar de alguns campos poderem ser omitidos e outros comprimidos, a especificação da compressão deixa a opção de parte desses campos serem mantidos integralmente. A Figura 5 mostra o que acontece com cada campo em um cabeçalho IPv6 na compressão IPHC, considerando o melhor caso, em que a máxima compressão é obtida.

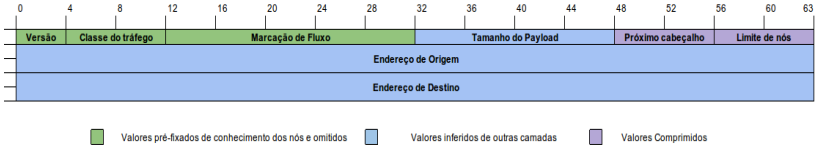


Figura 5 – Cabeçalho IPv6 e alterações promovidas na compressão IPHC 6LoWPAN

Fonte: elaborada pelo autor

Os campos comprimidos têm seus valores delimitados a certos intervalos ou passos definidos e são reduzidos a alguns bits no byte de *Dispatch* ou da compressão IPHC. A compressão IPHC utiliza o próprio *Dispatch* Byte para carregar informações, sem prejuízo à identificação do tipo de compressão. O menor tamanho do cabeçalho IPv6 é 40 bytes. A compressão do cabeçalho utilizando IPHC pode reduzir o seu tamanho a até 2 bytes: 1 de *Dispatch* IPHC e o outro com informações do cabeçalho comprimido que o segue. As alterações promovidas pela camada de adaptação 6LoWPAN com tal compressão são descritas em maiores detalhes na Tabela 1.

Os dados contidos nos bytes de *Dispatch* e IPHC contêm as definições

Tabela 1 – Compressão 6LoWPAN

Campo	Tamanho IPv6	Alteração
Versão	4 bits	Versão é sempre 6
Classe do tráfego	8 bits	Classe é sempre 0
Marcação do fluxo	20 bits	Marcação é sempre 0
Tamanho do Payload	16 bits	Inferido da camada de enlace ou do cabeçalho de fragmentação
Tipo do próximo cabeçalho	8 bits	Utiliza LOWPAN_NHC
Limite de saltos	8 bits	Fixo quando não há nós intermediários
Endereço de origem	128 bits	Inferido da camada de enlace
Endereço de destino	128 bits	Inferido da camada de enlace

do cabeçalho comprimido – quais campos seguem completos, quais são comprimidos e de que forma.

A compressão NHC permite que cabeçalhos das camadas superiores também sejam comprimidos. No caso do UDP, por exemplo, os próprios números de portas podem ser comprimidos caso sejam usados valores de um intervalo específico, assim como o *checksum* UDP. Os cabeçalhos UDP podem ser reduzidos de 8 bytes, no padrão IPv6, para apenas 1 byte no caso de maior compressão. Considerando que o UDP é o mais adequado em redes de baixa velocidade, dado o menor *overhead*, tal compressão permite ainda melhor aproveitamento da banda disponível. Mais detalhes no documento da RFC6282 (HUI; THUBERT, 2011).

3.3.2 Fragmentação

No IPv6 os enlaces precisam ter o MTU de pelo menos 1280 bytes. Isto significa que a rede precisa suportar no mínimo este tamanho de pacote para ser completamente compatível com redes IPv6. Em redes baseadas na IEEE 802.15.4 o tamanho do quadro na subcamada MAC é de no máximo 102 bytes. Para ser compatível com redes IPv6, portanto, o *gateway* deve ser capaz de fragmentar datagramas IPv6, assim como concatenar datagramas fragmentados em quadros da rede IEEE 802.15.4 IPv6. Tal fragmentação ocorre na camada de adaptação 6LoWPAN, entre a camada de rede e de enlace, não possuindo relação com algoritmos de fragmentação de camadas superiores.

Pacotes pequenos, que cabem inteiramente em um quadro da camada de enlace, não precisam ser fragmentados. Caso o datagrama IPv6 exceda determinado tamanho ele é fragmentado. Cada quadro da camada de enlace possui um cabeçalho de fragmentação. Todos os fragmentos do mesmo datagrama possuem o tamanho do pacote completo e uma *tag* de identificação do datagrama. Do segundo quadro até o último há ainda um contador, ou

offset, que identifica a posição do fragmento no datagrama.

A estrutura do cabeçalho de fragmentação para o primeiro pacote é exposta na Figura 6 e para os demais pacotes na Figura 7.

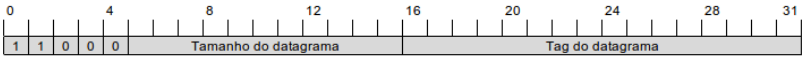


Figura 6 – Cabeçalho do primeiro fragmento
Fonte: elaborada pelo autor

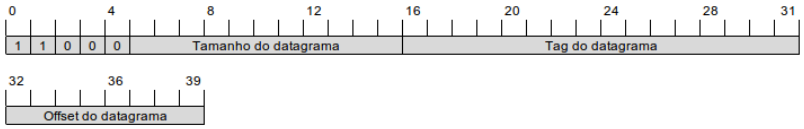


Figura 7 – Cabeçalho do segundo fragmento e subsequentes
Fonte: elaborada pelo autor

- **Tamanho do datagrama**: tamanho total do datagrama IP antes da fragmentação na camada de enlace. O mesmo valor é enviado em todos os fragmentos.
- ***Tag* do datagrama**: identifica a sequência de quadros de um mesmo datagrama IPv6. Deve ser incrementado a cada pacote enviado.
- ***Offset* do datagrama**: presente apenas no segundo fragmento e posteriores. Representa o *offset* do fragmento sendo enviado – valor deve ser múltiplo de 8.

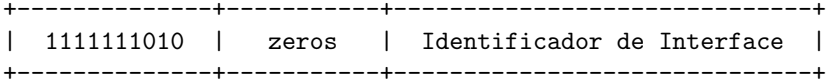
A especificação da fragmentação dos pacotes é detalhada na RFC4944 (MONTENEGRO et al., 2007).

3.3.3 Autoconfiguração de Endereços

O mecanismo de autoconfiguração de endereços de cada nó permite que os endereços das interfaces IPv6 sejam configurados sem intervenção manual em cada nó e sem a necessidade de servidores adicionais. Roteadores podem requerer configuração mínima.

Quando não há a presença de roteadores os nós podem gerar endereços locais, o que permite a comunicação entre nós da mesma rede. Duas situações são estabelecidas na geração do identificador de interface no *link* local, de acordo com o endereço IEEE 802.15.4 do dispositivo. No caso do endereço do nó ser longo, com 64 bits, a formação do identificador da interface se baseia no identificador EUI-64 conforme o padrão definido na RFC2464 (CRAWFORD, 1998), *IPv6 over Ethernet*. No caso do dispositivo utilizar endereço curto de 16 bits o identificador de interface se baseia em um pseudo-endereço de 48 bits com a estrutura formada da seguinte forma: 16 bits endereço da PAN : 16 zeros : 16 bits do endereço curto. O identificador da interface é formado então por estes 48 bits, novamente de acordo com a RFC2464.

O endereço de *link* local é formado a partir do identificador da interface seguindo a seguinte definição:



No caso de endereços globais o identificador de interface é utilizado em conjunto com um prefixo enviado pelo roteador em mensagens apropriadas. O nó pode decidir inclusive por solicitar tais mensagens na rede, evitando o atraso de esperar alguma mensagem enviada periodicamente pelo roteador. Ainda, em caso de não haverem roteadores, o serviço de DHCPv6 pode ser utilizado. A formação de endereços é especificada na RFC4862 (THOMSON; NARTEN; JINMEI, 2007) conforme mencionado anteriormente.

3.3.4 RPL

A RFC6550 (WINTER et al., 2012) define o RPL, um protocolo de roteamento para redes IPv6 desenvolvido com foco em LLNs. O RPL define mecanismos para tráfego multiponto para ponto, ponto para multiponto e ponto para ponto e é o padrão de roteamento em LLNs 6LoWPAN, apesar de não fazer parte do padrão 6LoWPAN em si. Redes 6LoWPAN são usualmente empregadas em cenários onde há centenas ou milhares de nós se comunicando em ambientes onde a quantidade de perdas pode ser alta e a disponibilidade de energia limitada. O RPL é pensado para funcionar mesmo com todos estes fatores presentes.

O RPL basicamente cria um grafo direto acíclico, denominado DODAG, tendo como base uma função objetivo e uma combinação de

métricas e restrições. Tais parâmetros podem ser específicos para cada rede, como por exemplo, a preferência por nós no caminho que não sejam operados a bateria, intensidade do sinal, dentre outros. O grafo é criado com base em informações trocadas entre os nós através de mensagens específicas. Após criado, tal grafo define a topologia da rede e os caminhos que os dados devem percorrer, sendo que um mesmo nó pode fazer parte de um ou mais grafos.

As regras definidas para a formação do grafo e a topologia resultante em si são determinantes no desempenho de uma rede e permitem que o protocolo seja adaptado para cada situação. Uma rede cujo meio seja ruidoso, mas na qual a disponibilidade de energia é ilimitada, pode se beneficiar disso e selecionar métricas que otimizem rotas em que a relação sinal-ruído é melhor. Uma mesma rede pode ter múltiplas topologias de roteamento ao mesmo tempo, podendo escolher deixar ativa a que mais se adapta aos requisitos do tráfego de dados em determinado momento.

É importante notar que o RPL opera na camada IP e portanto permite o roteamento entre múltiplos tipos de camadas de enlace, em contraste com outras formas de roteamento que operam em camadas inferiores, como na própria camada de enlace (VASSEUR et al., 2011).

3.4 CONSIDERAÇÕES FINAIS

As melhorias proporcionadas pela adoção do IPv6 são proporcionais à importância deste protocolo para a comunicação entre dispositivos em escala mundial. Além da necessidade do aumento de endereços, as facilidades e novas funcionalidades simplificam a Internet e a tornam mais confiável e completa. O padrão 6LoWPAN objetiva a integração de dispositivos simples e de poucos recursos à Internet, permitindo que, mesmo sob restrições, tais dispositivos sejam compatíveis com o protocolo IPv6. O fato de serem padrões abertos vai ao encontro dos objetivos do trabalho e do conceito de Internet das Coisas.

4 TRABALHOS RELACIONADOS

4.1 INTRODUÇÃO

A literatura recente sobre o padrão IEEE 802.15.4 é vasta e compreende trabalhos que vão desde análises sobre o desempenho dos protocolos CSMA/CA até o escalonamento de GTS. Há também alguns trabalhos que tratam da construção de nós em conformidade com esse padrão. Finalmente, há muitos trabalhos sobre sensores embarcados na Internet das Coisas. Com o objetivo de enfocar nos trabalhos relacionados a esta dissertação, nesse capítulo são considerados apenas aqueles trabalhos relacionados ao conceito de Internet das Coisas e ao padrão 6LoWPAN.

4.2 INTERNET DAS COISAS

O conceito de Internet das Coisas, de acordo com (ASHTON, 2009) foi cunhado por ele mesmo em uma palestra proferida em 1999. A ideia foi proposta tendo em mente a conexão entre a cadeia de fornecimento de uma importante empresa à Internet, sendo que a própria Internet era considerada novidade nesta década e estava nos seus estágios iniciais de crescimento. Inicialmente acreditava-se que o uso do RFID era obrigatório na Internet das Coisas, uma vez que seu uso permitia a integração e identificação de "todas as coisas" aos computadores. O conceito porém se tornou mais abrangente em termos de tecnologia com o passar do tempo. O termo se difundiu em 2005, quando a União Internacional de Telecomunicações, a ITU, publicou o primeiro relatório a respeito (GUSMEROLI et al., 2010).

Percebeu-se posteriormente que a aplicação de diversos conceitos e tecnologias é necessária para que a Internet das Coisas se torne realidade. (GUSMEROLI et al., 2010) percorre várias tecnologias de identificação, *web services*, comunicação, descoberta de redes, processamento de sinais, segurança e privacidade, entre outros, que de alguma forma podem ser utilizadas na integração de todas as coisas à Internet.

(ATZORI; IERA; MORABITO, 2010) estabelece três visões principais para o conceito de Internet das Coisas: uma visão orientada à Internet, uma visão orientada às coisas e uma visão semântica do conceito em si. O trabalho defende que diferentes organizações e pessoas em momentos distintos enxergam o conceito de uma das três perspectivas. Na perspectiva focada na Internet, foca-se na conectividade entre equipamentos e formas de endereçá-

los. A forma mais antiga de interpretar o conceito foca nas coisas, nas tecnologias para permitir que objetos do mundo real sejam tratados no mundo digital e integrados entre si. A visão semântica é fundamentada no fato de que a quantidade de novas tecnologias e dispositivos envolvidos na Internet do futuro é tão desafiadora que deve-se focar no significado das informações trocadas na tentativa de modelar e organizar os sistemas envolvidos em detrimento do detalhamento de cada dispositivo envolvido individualmente.

Recentemente o IEEE criou uma página na internet[1], organizou uma conferência[2] e criou um jornal[3], a ser lançado em 2014. O instituto tem papel fundamental na Internet das Coisas como um dos participantes ativos na criação e instituição de padrões. A importância de padrões é tratada em (DURST, 2013), que frisa a importância da existência de padrões ao compará-los com "tecidos de conexão", que permitem que tudo se conecte e interopere.

4.3 6LOWPAN

Além das RFCs que especificam os protocolos, há vasta literatura a respeito do uso e de implementações do 6LoWPAN disponíveis em código aberto, sem tratar de hardware específico, mas com análise das implementações e do protocolo em si.

O trabalho (MULLIGAN, 2007) é o primeiro a fazer referências ao 6LoWPAN - o seu autor é apontado como responsável pela criação do conceito. Nele, é detalhada a especificação em forma menos técnica que a RFC e são analisados futuros desafios e melhorias para o 6LoWPAN. Foca-se nas decisões tomadas ao longo do desenvolvimento do padrão e na aplicação dos conceitos de manipulação dos cabeçalhos para evitar a sobrecarga de dados de protocolo. Ao final do trabalho são apresentados desafios a serem enfrentados ao longo da adoção do 6LoWPAN.

A abordagem genérica do 6LoWPAN pode ser encontrada em diversos livros e artigos, com análises superficiais e breves do protocolo. O livro (SHELBY; BORMANN, 2009) é uma fonte que, apesar de relativamente desatualizada, fornece informação mais aprofundada, versando a respeito do posicionamento do protocolo no conceito de Internet das Coisas, as técnicas utilizadas na sua implementação e o detalhamento das características operacionais do protocolo.

A IPSO[4] é uma aliança de empresas incumbida de promover tecnolo-

[1] http://iot.ieee.org/
[2] http://sites.ieee.org/wf-iot/
[3] http://iot-journal.weebly.com/
[4] http://www.ipso-alliance.org

gias e dispositivos para redes IP, dentre os quais o 6LoWPAN e dispositivos compatíveis. O relatório técnico (HUI; CULLER; CHAKRABARTI, 2009) desta aliança é talvez a mais compacta e ao mesmo tempo completa referência a respeito do 6LoWPAN. O documento inicia justificando a adoção do IEEE 802.15.4 e as dificuldades iniciais em se adotar o IP sobre tal especificação - fato que levou fabricantes a utilizar protocolos proprietários e soluções de comunicação contemplando apenas camadas baixas do modelo OSI. Em seguida expõe que o 6LoWPAN alterou este cenário ao permitir o uso da especificação com os padrões de protocolos da Internet, utilizando o restante do relatório para detalhar a arquitetura, as alterações promovidas pela camada de adaptação e os recursos e vantagens do 6LoWPAN. O relatório finaliza expondo outros trabalhos relacionados a redes pessoais de baixo alcance e afirmando nas conclusões o quão atrativo é o uso de IP mesmo nos menores dispositivos.

Uma análise detalhada do desempenho das formas de compressão propostas na RFC6282, LOWPAN_IPHC e LOWPAN_NHC, e comparação com as técnicas de compressão originais, LOWPAN_HC1 e LOWPAN_HC2 definidas na RFC4944, é realizada em (LUDOVICI et al., 2009). O trabalho mostra como as formas de compressão mais recentes permitem ganhos significativos nos principais casos de utilização do protocolo. São analisados e medidos diferentes fatores na utilização dos diferentes tipos de compressão: consumo de corrente, tempo de transmissão e uso de memória. Mostrou-se que, apesar dos algoritmos de compressão mais recentes exigirem mais memória em sua implementação, as vantagens decorrentes dos recursos disponibilizados, o menor consumo de corrente e o menor tempo de transmissão os tornam mais adequados, justificando a adoção de tais algoritmos na especificação base do 6LoWPAN.

Há propostas para diferentes abordagens e modificações de trechos específicos da especificação original da 6LoWPAN, a RFC4944 (MONTENEGRO et al., 2007). Uma proposta interessante para uso de diferentes estratégias na camada MAC é o ContikiMAC (DUNKELS, 2011), elaborado pelo autor do sistema Contiki, Adam Dunkels. O artigo em si não propõe alterações no 6LoWPAN, mas sim uma nova forma de gerenciar o uso do rádio de forma que seja possível ainda maior economia de energia. O mecanismo consiste em manter o rádio desligado por cerca de 99% do tempo, ligando-o periodicamente para verificar a existência de dados e permanecendo ligado caso haja dados a serem recebidos. No caso da transmissão os dados são enviados em sucessivos intervalos para que os demais rádios utilizando tal estratégia possam detectar a transmissão e receber os dados adequadamente. Ao longo do trabalho a implementação do mecanismo é discutida e são realizados *benchmarks* para a sua validação. De acordo com o trabalho é possível

a economia de 10% a 80% de energia em uma rede de dispositivos, valor dependente dos parâmetros e configurações utilizados.

Em (SILVA; SILVA; BOAVIDA, 2009) é feita uma avaliação das três principais pilhas de software 6LoWPAN disponíveis. O trabalho analisa o consumo de memória RAM e ROM e a eficiência energética de cada uma das implementações. Dentre os resultados obtidos cita-se que o uso do IPv6, mesmo que combinado com a camada de adaptação 6LoWPAN, resulta em maior consumo energético em relação a protocolos mais simples, como era esperado. Ele conclui, porém, que em duas das implementações analisadas a diferença é diminuta e de fato já permite o uso do padrão 6LoWPAN em dispositivos pequenos e de poucos recursos.

Em se tratando da implementação de nós compatíveis com a especificação IEEE 802.15.4, e que possam fazer parte de redes 6LoWPAN, grande quantidade de dispositivos estão disponíveis comercialmente. Uma lista dos principais nós está disponível na página da Internet denominada *The Sensor Network Museum*[5]. Algumas poucas referências de trabalhos cujo resultado incluía o desenvolvimento de novos nós de comunicação foram encontradas porém o autor desta dissertação não obteve acesso aos textos completos.

Os *gateways* entre as LLNs 6LoWPAN e outros tipos de redes, dispositivos que permitem a aplicação do conceito de Internet das Coisas na prática, possuem menos implementações disponíveis. Trabalhos como o de (JIANG et al., 2009), a respeito de monitoramento de energia, e (MAYER; FRITSCHE, 2006), que trata da coleta de informações médicas de pacientes em um ambiente hospitalar, são citados frequentemente como referências, porém focam nas suas aplicações, sem entrar nos detalhes dos *gateways* 6LoWPAN desenvolvidos e utilizados.

Em (SINNIAH et al., 2012) é realizado o design e avaliação de desempenho fim-a-fim de um *gateway* 6LoWPAN. O *gateway* é constituído por um computador de mesa conectado a nós que executam o Sistema Operacional FreeRTOS[6]. Tal hardware permite a avaliação do uso do 6LoWPAN porém se distancia da proposta do atual trabalho em termos de obter um *gateway* com características de sistema embarcado, de tamanho reduzido, e condizente com a simplicidade oferecida por redes sem fio baseadas na IEEE 802.15.4. O *gateway* implementado é validado através do acesso a dados de sensores na rede 6LoWPAN em uma arquitetura cliente-servidor.

Outro *gateway* composto por um computador – neste caso baseado no

[5] http://www.snm.ethz.ch

[6] http://www.freertos.org

Sistema Operacional Mac OS[7] – conectado a um nó comercial é proposto em (CAMPOS et al., 2011). O nó compatível com IEEE 802.15.4 utilizado é o Iris[8], rodando o Sistema Operacional TinyOS. Este *gateway* implementa uma ponte interna que permite que dispositivos com endereço utilizando IPv4 também façam parte da rede.

O projeto 6LBR (CETIC, 2013) implementa um *gateway* 6LoWPAN baseado no Contiki capaz de executar em qualquer estação rodando Linux que possua conectado um nó IEEE 802.15.4 via USB. São fornecidas referências do sistema funcionando no computador Raspberry Pi[9] assim como na própria BeagleBone, utilizada neste projeto, em conjunto com um dispositivo TelosB [10].

4.4 CONSIDERAÇÕES SOBRE OS TRABALHOS

A adoção do 6LoWPAN em redes de sensores e atuadores sem fio e a aplicação do conceito de Internet das Coisas vêm crescendo. A quantidade de referências disponíveis *online* no início deste trabalho era bastante inferior à quantidade encontrada no momento da finalização deste documento. O aumento substancial de referências é uma evidência do crescimento do uso desse padrão.

O conceito de Internet das Coisas, apesar de já possuir mais de uma década, ainda dá margem para diferentes interpretações ao longo dos diferentes trabalhos analisados. As principais tecnologias envolvidas no conceito são conhecidas, porém claramente ainda há muito terreno a ser explorado. Os artigos e livros consultados mostram muitas previsões, estimativas e teorizam a respeito do futuro da Internet das Coisas, porém dadas as incertezas os esforços principais se concentram hoje na definição de padrões e na viabilização de tecnologias que permitam que o conceito ganhe ainda mais força no futuro.

Apesar de figurar como uma das tecnologias presentes na Internet das Coisas, o 6LoWPAN não recebe a devida atenção nos trabalhos que tratam o conceito de forma generalista. Considerando que o 6LoWPAN atualmente consiste em uma das soluções mais promissoras para integrar dispositivos de poucos recursos diretamente à Internet, sua importância foi certamente menosprezada em tais trabalhos - uma das possibilidades é o fato do 6LoWPAN estar ganhando notoriedade apenas recentemente.

[7] http://www.apple.com/osx/
[8] http://www.xbow.com
[9] http://www.raspberrypi.org/
[10] http://www.xbow.com

Os trabalhos que analisam o padrão 6LoWPAN e as suas principais implementações, como (SILVA; SILVA; BOAVIDA, 2009) e (LUDOVICI et al., 2009), são bastante objetivos e mostram a viabilidade do uso deste padrão. Deixam claro também que apesar de boa parte do padrão possuir referências de uso e implementações de software ainda há vários recursos a serem explorados, principalmente em se tratando da implementação de redes *mesh*.

As análises mais aprofundadas do protocolo se concentram em estudos teóricos e na análise da eficiência energética e do tamanho do código das implementações 6LoWPAN. O impacto comparativo em relação ao uso do IPv6 puro na prática, porém, é pouco explorado. Tal fato influenciou a realização deste trabalho, que procura mostrar o impacto prático dos benefícios promovidos pelo uso da camada de adaptação definida no 6LoWPAN.

No que diz respeito à implementação de *gateways*, (SINNIAH et al., 2012) e (CAMPOS et al., 2011) se focam no atendimento ao padrão, porém sem preocupação com os resultados em termos de hardware e a possibilidade de aproveitamento dos dispositivos desenvolvidos em aplicações de campo. Desenvolver equipamentos considerando as restrições comuns a estes tipos de redes inclui levar em consideração limitações de custo, restrições dimensionais e a necessidade de simplicidade de configuração, preocupações que não estavam no foco destes trabalhos.

O projeto 6LBR (CETIC, 2013) está mais próximo de uma situação real, ao permitir a compatibilidade com *dongles* USB conectados a computadores com Linux. Ainda assim nenhuma referência foi encontrada de algum dispositivo que cumpra o conceito de sistema embarcado ao extremo, ou seja, que execute apenas a função de *gateway*, com foco em simplicidade, redução de custos e de tamanho. Uma das dificuldades para a maior adoção do 6LoWPAN consiste justamente na ausência deste tipo de dispositivo. Neste trabalho, apesar do uso de um dispositivo mais robusto rodando o Sistema Operacional Linux, os nós desenvolvidos e suas interfaces permitem futuro desenvolvimento de dispositivo *gateway* integrado e dedicado. Esta possibilidade consistiu em um dos requisitos principais no planejamento do dispositivo.

4.5 CONSIDERAÇÕES FINAIS

A Internet das Coisas é um conceito novo. O 6LoWPAN é uma tecnologia nova. Trabalhos vêm surgindo, porém a quantidade de referências ainda é pequena se comparada com outros temas mais difundidos. Ao se

pesquisar por novos trabalhos e ao analisá-los, nota-se a grande quantidade de conceitos pouco aplicados e tecnologias pouco exploradas. Ao mesmo tempo que isto dificulta de certa forma novas pesquisas dada a escassez de referências, abre muitas possibilidades para novos trabalhos. Uma conclusão objetiva após a análise dos trabalhos é que a Internet das Coisas pode estar nos seus princípios, mas as tecnologias que a tornam viável já estão em estágio bastante avançado de desenvolvimento, tornando paulatinamente possível a integração de dispositivos cada vez mais simples à Internet.

5 PROJETO E IMPLEMENTAÇÃO DOS NÓS IEEE 802.15.4 E DO *GATEWAY* 6LOWPAN

5.1 INTRODUÇÃO

Inserir dispositivos na Internet das Coisas exige que estes possuam interfaces de comunicação e software adequados para tal. Neste trabalho foram projetados e implementados dispositivos modulares que permitem que equipamentos possam ser integrados à Internet de forma simples e direta. Módulos para adicionar interfaces de comunicação sem fio 6LoWPAN a dispositivos e um *gateway* foram desenvolvidos ao longo do trabalho e serão descritos nas seções seguintes.

5.2 NÓS IEEE 802.15.4

Os nós da rede sem fio baseada no padrão 6LoWPAN foram projetados e desenvolvidos no contexto deste trabalho. Uma rede sem fio composta por estes nós foi implementada inicialmente sem comunicação com dispositivos externos. O objetivo de tal implementação é a validação dos dispositivos sem fio, tanto em termos de hardware como de software, sem a influência do *gateway* e de redes externas.

Além de permitir o foco no desenvolvimento dos módulos e a validação do hardware, facilitando a detecção de possíveis erros, a decisão por implementar primeiro uma rede local com os nós permitiu que o autor se familiarizasse com os dispositivos antes da integração com outras redes. A própria rede isolada por si só abre inúmeras portas para o desenvolvimento de aplicações de alto nível.

5.2.1 Requisitos

No projeto dos dispositivos foram inicialmente definidos os requisitos desejados, em alto nível, do ponto de vista do que se deseja para o produto final, sem preocupação com tecnologias utilizadas.

1. **Utilizar padrão de comunicação sem fio aberto e compatível com a Internet**: em se tratando de compatibilidade com a Internet das Coisas o uso de padrões abertos é fundamental, uma vez que os protocolos que dominam a Internet possuem especificação aberta. Utilizar protocolos

proprietários limita a conectividade do dispositivo e possibilidade de uso em diversos cenários.

2. **Dispositivos devem possuir alcance de no mínimo 40 metros**: considerando-se plantas industriais tal distância é justificada. Para distâncias maiores outros projetos podem contemplar antenas de maior potência.

3. **Nós devem ser operados a bateria**: flexibilidade na instalação por não serem necessários cabos de força.

4. **A arquitetura dos nós deve ser modular**: o desenvolvimento orientado a módulos evita o retrabalho em cada produto na incorporação dos circuitos e software de comunicação.

5. **O módulo deve possuir dimensões reduzidas**: área máxima do módulo de 5 cm x 5 cm, viabilizando o uso em equipamentos de pequeno porte.

Em seguida foram avaliadas as tecnologias e protocolos disponíveis que atendem os requisitos desejados. Em se tratando de tecnologias para transmissão de dados sem fio no alcance estipulado e com foco em dispositivos operados a bateria os principais padrões são:

- **Bluetooth®**[1]: alcance de até 100 metros, opera na banda de 2.4GHz. Mantido por um consórcio de milhares de empresas é amplamente adotado em redes sem fio pessoais. Define desde a camada física até a camada de aplicação no modelo OSI. A tecnologia é patenteada e seu uso exige o pagamento de licenças.

- ***Certified Wireless* USB**[2]: sucessor sem fio do USB convencional, o qual é considerado padrão *de facto*. Emprega tecnologia UWB e opera nas frequências de 3.1 GHz a 10.6 GHz, permitindo taxas de transferência de até 480 Mbits/s com alcance limitado de até 10 metros. Para que um dispositivo seja considerado compatível e possa exibir a logomarca da tecnologia o pagamento de taxas é necessário.

- **IEEE 802.15.4**[3]: especifica as camadas de baixo nível para comunicação sem fio de dispositivos em redes de curto alcance. É amplamente adotada em dispositivo com baixa disponibilidades de recursos e serve como base para padrões como WirelessHART®, Zigbee® e 6LoWPAN.

[1] http://www.bluetooth.com
[2] http://www.usb.org/developers/wusb
[3] http://www.ieee802.org/15/pub/TG4.html

- **WirelessHART®**[4]: especificação voltada para o ambiente industrial, alternativa sem fio à tecnologia cabeada HART. Baseada na IEEE 802.15.4 opera na frequência de 2.4 GHz. Não é necessário licenciamento para o uso deste padrão.

- **Zigbee®**[5]: voltado a aplicações de baixo consumo e baixa taxa de transferência o Zigbee® também é baseado na IEEE 802.15.4, especificando as camadas acima das definidas neste padrão. É bastante utilizado em aplicações de automação residencial, gerenciado pela Zigbee® *Alliance* e exige pagamento de licença para que um dispositivo seja considerado compatível pela organização.

- **6LoWPAN**: é um padrão que permite o envio de datagramas IPv6 sobre as camadas de baixo nível definidas na especificação IEEE 802.15.4, agindo como camada de adaptação entre a camada de transporte do modelo TCP/IP e a subcamada de enlace e camada física definida na especificação. É um padrão aberto e não exige uso de licenças.

Todos os padrões mencionados são bastante utilizados e possuem vantagens específicas, voltadas para a especificidade das aplicações alvo de cada um. Levando em consideração os requisitos definidos no projeto foi definido o padrão a ser utilizado. Em termos de alcance eliminou-se inicialmente a possibilidade de uso do Wireless USB. O desejo de um padrão aberto e sem a necessidade de licenciamento impede o uso do Bluetooth® e Zigbee®. Sobram as alternativas baseadas na especificação IEEE 802.15.4, WirelessHART® e 6LoWPAN, além do uso da própria especificação isolada.

O requisito principal na definição do padrão a ser utilizado, e que por si só deixa apenas uma opção de escolha, reside na compatibilidade com a Internet. O WirelessHART não possui compatibilidade com a pilha TCP/IP, protocolo padrão da Internet, e a especificação IEEE 802.15.4 define apenas camadas abaixo da camada de rede. A escolha pelo padrão 6LoWPAN é, portanto, lógica e direta.

5.2.2 Hardware

O hardware dos nós de comunicação foi desenvolvido pelo mestrando na empresa Boreste Sistemas Embarcados. A Boreste é especialista no desenvolvimento de sistemas embarcados, da concepção à produção. Os dispositivos desenvolvidos na empresa têm características fortemente

[4] http://www.hartcomm.org/protocol/wihart/wireless_technology.html
[5] http://www.zigbee.org

modulares e podem se beneficiar tanto dos nós como do *gateway* implementados no decorrer deste trabalho, permitindo a integração dos mais diversos equipamentos via tecnologia sem fio focados no baixo consumo. Os dispositivos foram projetados tendo em mente a ideia de nós de comunicação genéricos, operados a bateria, com interface padrão desenvolvida pela Boreste para acoplamento de possíveis módulos de sensores e atuadores.

Foi considerado o uso de módulos prontos, como o Digi XBee®PRO[6] ou os módulos da série Atmel ZigBit[7], porém o custo elevado destes módulos, o fato do foco da empresa Boreste ser o desenvolvimento de projetos em eletrônica embarcada e o objetivo do projeto ser possuir um módulo próprio que permita a integração de dispositivos à Internet das Coisas são fatores que direcionaram à construção de um módulo próprio.

Cada módulo deve ser microprocessado e possuir um transceptor de rádio compatível com o padrão IEEE 802.15.4. Diversas opções de transceptores foram consideradas para o projeto dos nós, conforme a Tabela 2. Os preços na tabela consideram a média na compra de 10 unidades no momento da pesquisa destes componentes no contexto do projeto. Todos os dispositivos selecionados operam na frequência 2.4GHz.

Tabela 2 – Comparação entre transceptores IEEE 802.15.4 considerados no projeto

Dispositivo	**Pilhas de Software**	**Encapsulamento**	**Preço**
Atmel AT86RF231	IEEE 802.15.4 MAC[8], BitCloud[9], ZigBee RF4CE[10]	QFN-32	US$ 3.15
TI CC2520	TI-MAC[11], Z-Stack[12]	VQFN-28	US$ 5.14
Freescale MC13202	Freescale BeeStack[13], IEEE 802.15.4 MAC	QFN-32	US$ 4.11
Microchip MRF24J40	MiWi[14] Zigbee®[15]	QFN-40	US$ 3.24

Além dos transceptores mencionados acima é necessário um microcontrolador para comandar o módulo. Tal microcontrolador deve possuir a pilha de software necessária para a comunicação na rede 6LoWPAN, isolando o restante da aplicação - o hardware o qual terá o módulo acoplado

[6] http://www.digi.com/products/wireless-wired-embedded-solutions/zigbee-rf-modules/point-multipoint-rfmodules/xbee-series1-module

[7] http://www.atmel.com/products/microcontrollers/wireless/modules.aspx

[8] http://www.atmel.com/tools/IEEE802_15_4MAC.aspx

[9] http://www.atmel.com/dyn/products/tools_card.asp?tool_id=4495

[10] http://www.atmel.com/dyn/products/tools_card.asp?tool_id=4712

[11] http://www.ti.com/tool/timac

[12] http://www.ti.com/tool/z-stack

[13] http://www.freescale.com/files/rf_if/doc/ref_manual/BSSRM.pdf

[14] http://www.microchip.com/miwi/

[15] http://www.microchip.com/zigbee/

- de detalhes da comunicação. Na pesquisa por transceptores encontraram-se *chips* que possuem ambos microcontrolador e transceptor integrados. A economia em termos de tamanho é significativa, e uma vez que um dos principais requisitos diz respeito à miniaturização do módulo, tais componentes foram avaliados. Eles são denominados SoCs, do inglês *System on a Chip*, e são exibidos na Tabela 3.

Tabela 3 – Comparação entre SoCs considerados no projeto

Dispositivo	**Componentes Internos**	**Pilha de Software**	**Encapsulamento**	**Preço**
Ember - EM350/EM357	ARM Cortex-M3 + *transceiver*	EmberZNet PRO - Zigbee PRO[16]	QFN-48	US$ 6
Atmel ATmega128RFA1	SoC: ATmega1281 8-bit MCU + AT86RF231 *transceiver*	6LoWPAN, BitCloud, ZigBee® RF4CE	QFN-64	US$ 8.8
Analog Devices - ADuCRF	ARM Cortex-M3 MCU + *transceiver*	BitCloud - ZigBee® PRO	LFCSP-64	Ainda não disponível para venda

Considerando o ganho de tamanho promovido pelo uso de uma solução integrada, somado à redução de custo decorrente, a decisão se concentrou na escolha entre os três dispositivos integrados. O fato de não estar disponível para venda no início do projeto tornou a opção da Analog Devices inviável. Os dispositivos da família Ember EM35X da SiLabs[17] são direcionados ao uso do protocolo ZigBee®. A ausência de pilha suportada para 6LoWPAN em tais dispositivos e a alta disponibilidade de suporte em se tratando do ATmega128RFA1 acabou tornando certa a decisão pela solução integrada da Atmel. Além de possuir suporte ao 6LoWPAN e várias outras pilhas de software, facilitando o desenvolvimento, a documentação fornecida é vasta e completa.

Uma diferença significativa entre o ATmega128RFA1 e os demais é o fato deste *chip* ser baseado na arquitetura AVR de 8 bits, enquanto os demais possuem base na arquitetura Cortex-M3, de 32 bits, mais avançada e recente. Uma vez que se deseja um dispositivo modular, dedicado apenas à comunicação, o uso de uma arquitetura mais complexa resulta em maior consumo e disponibilidade de recursos que não são utilizados nas aplicações comuns, aumentando os custos sem necessidade. A opção pela arquitetura mais simples traz pouco prejuízo de funcionalidade a um custo justificável. Somado isto ao fato do dispositivo em si possuir também características úteis como o CSMA/CA, confirmação e retransmissão de pacotes automáticos,

[16] http://www.silabs.com/products/wireless/zigbee/Pages/zigbee-software.aspx
[17] www.silabs.com

tudo incluso em um encapsulamento QFN-64 de tamanho reduzido, tem-se a decisão pelo ATmega128RFA1.

5.2.2.1 AVR-ATMEGA128RFA1

O módulo de rádio desenvolvido é baseado no componente Atmel ATmega128RFA1 que combina, em um mesmo *chip*, um microcontrolador RISC de 8 bits da família AVR e um *transceiver* 2.4 GHz compatível com redes IEEE 802.15.4. Ele é derivado do microcontrolador ATMEGA1281 e do *transceiver* AT86RF231. O microcontrolador Atmel ATMEGA1281 faz parte de uma linha simples e de baixo consumo de microcontroladores de 8 bits.

O *chip* ATmega128RFA1 foi selecionado por suas características, dentre as quais pode-se citar:

- **Apresenta baixo consumo**: corrente máxima de 18.6 mA quando transmite dados na rede e cerca de 4 mA quando ocioso. Possui modos de baixo consumo que podem chegar a correntes inferiores a 1 uA.
- **Encapsulamento reduzido**: permite a construção de um módulo de rádio pequeno para ser utilizado em conjunto com outros produtos de tamanho reduzido.
- **Integração de funcionalidades**: facilidades na configuração do software e simplicidade de hardware. O fato de possuir microcontrolador e *transceiver* integrados diminui consideravelmente o tamanho do componente e o seu *footprint* na placa.
- **Disponibilidade de recursos de software**: disponibilidade de bibliotecas da própria fabricante e compatibilidade com Sistemas Operacionais de código aberto disponíveis.
- **Memórias disponíveis de acordo com as aplicações alvo**: memória SRAM de 16KiB, memória Flash de 128KiB e memória EEPROM de 4KiB.
- **Frequência do clock principal de até 16MHz**: ajustável para permitir menor consumo, também de acordo com as aplicações pretendidas para o módulo desenvolvido.

Outro fator que foi importante na escolha do processador foi um trabalho prévio (AFANASYEV et al., 2010) que mostrava a adequação de seus módulos de rádio para a implementação do 6LoWPAN. Nesse trabalho

foram realizados testes com módulos de rádio distintos. Segundo o autor, o rádio AT86RF230 apresentou bom desempenho, se mostrando adequado para ser utilizado em redes IP em condições de tráfego heterogêneo, sendo factível o seu uso em redes sem fio de baixo consumo. O rádio AT86RF231 é semelhante ao AT86RF230, adicionando alguns recursos extras como a possibilidade de uso de velocidades maiores e a presença de um gerador de números aleatórios por hardware, entre outras novas funcionalidades.

O *chip* escolhido fornece funções e facilidades para redes IEEE 802.15.4, como por exemplo assistência via hardware para confirmação e repetição automáticas, computação de CRC-16, entre outras. Informações detalhadas a respeito do dispositivo podem ser encontradas em (ATMEL, 2013).

5.2.3 Software

Uma decisão por desenvolver todo o software para o funcionamento do dispositivo incluiria a programação dos seguintes itens:

- *Drivers* de baixo nível do microcontrolador
- *Drivers* de baixo nível do rádio
- Bibliotecas básicas de interface e comunicação
- Pilha de software IEEE 802.15.4
- Pilha de rede TCP/IP
- Implementação da camada de adaptação 6LoWPAN

Optou-se por procurar soluções de software já disponíveis na Internet. A própria Atmel possui *drivers* e bibliotecas disponíveis para o uso com o dispositivo. O código de suporte para 6LoWPAN da fabricante, porém, foi tornado obsoleto ao longo do desenvolvimento deste projeto. Alternativas de software foram procuradas para evitar a necessidade de implementação do software desde o baixo nível. Dois sistemas operacionais foram encontrados em pesquisas na Internet, ambos bastante utilizados e com grande quantidade de referências na Internet: o Contiki e o TinyOS[18]. Os dois sistemas operacionais foram testados e exemplos de uso analisados.

O fato de ter sido desenvolvido tendo em mente o conceito de Internet das Coisas, de ser apoiado pela própria Atmel, de ter suporte ao dispositivo

[18] www.tinyos.net

ATmega128RFA1 (aprimorado no TinyOS no decorrer da execução deste trabalho) e de ser implementado em linguagem C tornou o Contiki a opção mais adequada para este trabalho. Além disto, o Contiki possui implementação das técnicas de compressão de cabeçalhos LOWPAN_IPHC e LOWPAN_NHC, definidas na RFC6282 (HUI; THUBERT, 2011), as quais apresentam melhor resultado que as técnicas especificadas na RFC4944.

A pilha de rede completa de cada dispositivo, resumindo o que foi exposto até aqui, é representada na Figura 8.

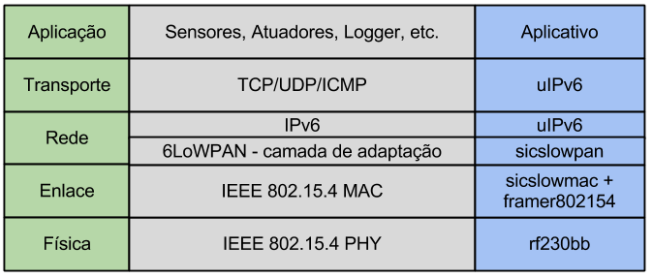


Figura 8 – Relação entre OSI, pilha utilizada no projeto e pilha no Contiki
Fonte: elaborada pelo autor

5.2.3.1 Contiki

O Contiki (THE CONTIKI PROJECT AND ITS CONTRIBUTORS, 2003), desenvolvido inicialmente por Adam Dunkels no Instituto Sueco de Ciências da Computação, é um Sistema Operacional de código aberto desenvolvido tendo como objetivo aplicar o conceito de Internet das Coisas. Diversas instituições e empresas colaboraram para o desenvolvimento do sistema, como Atmel, Cisco, SAP e outras. A implementação do sistema é bastante leve e otimizada, permitindo a criação de aplicativos de apenas alguns kilobytes com recursos multitarefa e comunicação TCP/IP, além de possuir implementados os *drivers* de baixo nível do microcontrolador selecionado neste trabalho e vários outros.

A combinação do hardware do *chip* ATmega128RFA1 e suas características de baixo consumo com o código otimizado para economia de energia do Contiki permitem a criação de dispositivos que, mesmo alimentados com baterias simples, possam se comunicar utilizando TCP/IP em redes sem fio. A economia de energia advém basicamente dos modos de baixo consumo do microcontrolador e dos mecanismos de *duty cycling* do

rádio (RDC) da comunicação no Contiki.

O Contiki fornece suporte de software em pontos chave do contexto do projeto. A lista abaixo informa algumas das características mais importantes em cada um dos níveis de implementação do Contiki:

- **Interface com outros dispositivos de rede**: suporte a protocolos SLIP e PPP.
- **TCP/IP**: suporte a procotolos TCP, UDP e ICMP. Utiliza a pilha uIP – menor pilha TCP/IP validada pela Cisco (DUNKELS et al., 2008).
- **6LoWPAN**: implementação da camada de adaptação do 6LoWPAN, incluídos algoritmos de compressão de cabeçalho, conversão de endereços e fragmentação.
- **IEEE 802.15.4**: camada física (PHY) e múltiplas implementações da subcamada MAC.
- **Suporte para desenvolvimento de aplicativo**: m*ultitasking* (protothreads), criação de sockets estilo BSD (protosockets) e implementação da biblioteca C básica.
- **Drivers ATmega128RFA1**: *drivers* de baixo nível do dispositivo e dos periféricos prontos no Sistema Operacional de versão 2.6.

5.2.3.2 Suporte à especificação IEEE 802.15.4

Ao ser concebido para executar no menor dos dispositivos, o Contiki é baseado na especificação IEEE 802.15.4 nas camadas inferiores do modelo OSI. O suporte a tal especificação, porém, é incompleto. Apenas o modo sem *beacon* é suportado e por consequência o uso de GTS também não é coberto. Neste trabalho foi utilizado o modo sem *beacon* para troca de dados, sendo o controle de acesso ao meio via CSMA/CA realizado pelo hardware. O Contiki suporta a fragmentação definida na RFC4944 integralmente e também a compressão especificada na RFC6282.

5.2.3.3 uIPv6

O Contiki possui embutido em seu software a pilha uIP – ou micro IP. Esta é uma pilha completa TCP/IP otimizada para dispositivos com pouco poder de processamento e pouca memória. Sua versão IPv6, denominada

uIPv6, foi desenvolvida em conjunto pela Atmel, Cisco e pelo Instituto Sueco de Ciências da Computação e é considerada a menor implementação compatível com esse novo protocolo a ser aprovada na fase 1 da validação do IPv6 (DUNKELS et al., 2008).

Protocolos bastante comuns em WPANs, como o Zigbee® e o WirelessHART®, possuem código proprietário e impedem a interoperabilidade entre outros dispositivos e mesmo a conexão com dispositivos em redes Ethernet comuns. A simplicidade da pilha uIPv6 no Contiki é umas das responsáveis pela possibilidade do uso do conceito da Internet das Coisas, onde todos os dispositivos, por menor que sejam, possam fazer parte de uma rede IPv6, podendo se comunicar inclusive com a Internet.

Somando isso ao fato de ser um padrão aberto, a adoção do IPv6 em dispositivos os tornam mais compatíveis com as diversas redes e equipamento existentes, além de diminuir a dependência de empresas e dispositivos específicos. Vale lembrar que no caso do Zigbee®, por exemplo, a certificação de compatibilidade de um produto exige pagamento, caso contrário não se pode afirmar de forma oficial que tal produto é compatível com o padrão Zigbee®.

Toda a implementação da pilha uIPv6 é baseada em um *buffer* único, tornando mais eficiente o uso da memória RAM. O *buffer* utilizado é global e seu tamanho define o máximo tamanho de pacotes com os quais o dispositivo pode lidar. Por possuir um *buffer* único, uma aplicação deve processar dados no *buffer* antes que novos dados sejam recebidos – tal processamento pode ser simplesmente passar para um *buffer* próprio da aplicação caso haja memória suficiente disponível.

Existem duas APIs para acesso às funções da pilha uIPv6: as *protosockets*, interface de *sockets* semelhante à interface BSD, e a API de acesso direto (*raw*), baseada em eventos, onde funções da aplicação são chamadas em respostas a determinados eventos. As protosockets suportam apenas TCP. As funções de acesso direto exigem menos memória e são utilizadas no caso de outros protocolos, como o UDP.

A dinâmica do código é bastante simplificada. Um processo principal gerencia um vetor de conexões. A cada chamada deste processo uma conexão é processada. Isto difere de implementações maiores, em que uma tarefa é criada para gerenciar cada conexão. Tal fato aliado ao *buffer* único são fundamentais para a baixa exigência de recursos da pilha. Apesar disto os protocolos TCP, UDP, IPv4 e IPv6 são suportados, assim como os protocolos de mensagens de gerência da rede, ICMP e ICMPv6.

O documento original de apresentação da pilha uIPv6 (DUNKELS et al., 2008) data de 2003 e não reflete o estado atual de implementação da mesma, apesar de ser fonte para informações gerais do projeto. Informações

aqui descritas foram extraídas de tal documento e da própria análise do código disponível no Contiki. A pilha uIPv6 é regida por licença de código aberto – seu uso é permitido em aplicações comerciais.

5.2.3.4 Configuração da pilha de comunicação

O Contiki permite que diferentes técnicas sejam utilizadas em cada camada da comunicação. Entenda-se por camadas neste contexto aquelas definidas pelo próprio Contiki que, apesar de terem semelhanças com o modelo OSI, possuem características específicas no contexto do Sistema Operacional. A Figura 9 contém a divisão em camadas do Contiki e as respectivas configurações para cada camada utilizada no projeto.

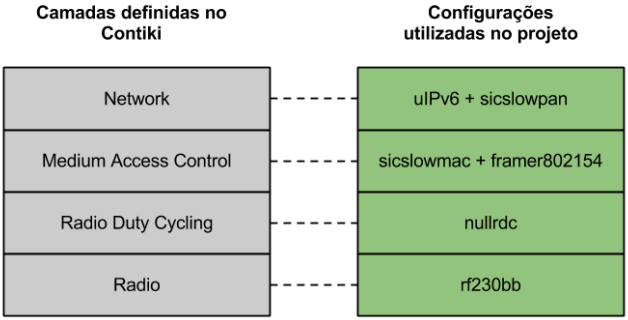


Figura 9 – Camadas Contiki
Fonte: elaborada pelo autor

Três opções são possíveis na camada de rede: a pilha uIP, compatível com TCP/IP IPv4, a pilha uIPv6, compatível com IPv6, e a pilha RIME (DUNKELS, 2007), específica do Contiki para quando o *overhead* das pilhas mencionadas anteriormente for considerado proibitivo para determinada implementação. No contexto deste trabalho foi utilizada a pilha uIPv6 com a camada de adaptação 6LoWPAN – no sistema as referências a tal configuração são feitas pelo nome *sicslowpan*. Apesar de a pilha RIME não ser utilizada, a própria implementação da camada 6LoWPAN se beneficia internamente do gerenciamento de *buffers* dessa pilha e algumas outras funcionalidades, mas que porém não influenciam no padrão TCP/IP e 6LoWPAN.

Na camada MAC tem-se duas opções: sem controle de acesso ao meio, na qual os dados simplesmente são enviados sem verificação do canal,

ou a opção CSMA com prevenção de colisão (CSMA/CA), que realiza o procedimento de CCA para verificação de canal vazio e evita colisões avisando aos demais nós de sua intenção de transmitir, determinando um tempo de *back-off* antes de retransmissão. Neste trabalho, técnicas de CSMA/CA são utilizadas, porém não por software. Isso porque o próprio *chip* selecionado ATmega128RFA1 possui CSMA/CA, controle de auto-repetição e autoconfirmação (ACK) implementados em hardware, dispensando tal controle pelo Sistema Operacional, o qual apenas se encarrega de configurar tais recursos adequadamente. Em termos de implementação tal camada foi configurada como *sicslowmac*, na nomenclatura do Contiki, que apenas organiza os quadros conforme a especificação IEEE 802.15.4, porém sem realizar o controle de acesso ao meio e de retransmissão, o qual é deixado para o *driver* de rádio via recursos de hardware.

A camada RDC é responsável pelo chaveamento do rádio, procurando reduzir o tempo em que o rádio permanece ligado sem comprometer a comunicação, permitindo economia de energia. Há mais opções de configuração disponíveis nesta camada: ContikiMAC, X-MAC, CX-MAC, e nullrdc. O ContikiMAC (DUNKELS, 2011) permite que o rádio permaneça desligado até 99% do tempo, acordando periodicamente para verificação de dados disponíveis para o nó na WPAN. Se a transmissão de um pacote é detectada durante o período em que o rádio está ligado, o mesmo é mantido ligado para recepção do pacote. O ContikiMAC foi inspirado no X-MAC e em outras implementações anteriores. Informações a respeito do X-MAC podem ser encontradas em (BUETTNER et al., 2006).

CX-MAC, ou Contiki X-MAC, é uma implementação do X-MAC específica do Contiki com algumas características adicionadas. Por último, *nullrdc* significa que nenhum recurso de *Radio Duty Cycling* será utilizado. Neste projeto não foi utilizado recurso de *Radio Duty Cycling* em testes regulares pois isto permitiu a comparação de forma mais clara entre usar ou não usar os recursos de compressão da camada de adaptação 6LoWPAN.

Na camada de rádio foi utilizado o *driver* para o rádio do ATmega128RFA1. Tal *driver* é referido como *rf230bb* na implementação do Contiki e, apesar do nome, também funciona com dispositivos AT86RF231.

5.2.4 Implementação

Cada dispositivo de comunicação é composto por um módulo de rádio, denominado Boreste SHIP ATmega128RFA1, e uma base, descritos em blocos na Figura 10.

O modelo tridimensional do dispositivo desenvolvido, incluída

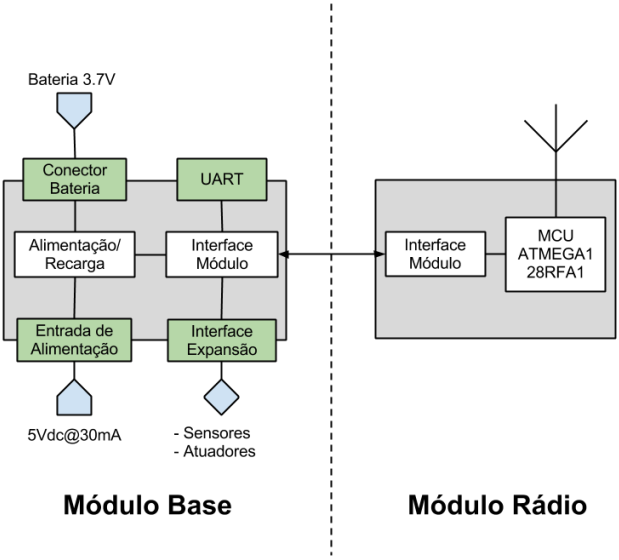


Figura 10 – Blocos funcionais do dispositivo
Fonte: elaborada pelo autor

solução de gabinete e bateria, é mostrado na Figura 11. O dispositivo pronto e montado pode ser visto na foto da Figura 12. O módulo de rádio desconectado da base pode ser visualizado na foto da Figura 13.

A interface de expansão possui interface Boreste 3SI que permite a conexão de outros dispositivos da empresa, como *loggers* e mesmo sensores e atuadores. A arquitetura modular do rádio permite que ele possa também ser conectado a outros equipamentos que contemplem a conexão SHIP Boreste, adicionando a tais dispositivos funcionalidades de comunicação 6LoWPAN, ou mesmo outros padrões baseados na especificação IEEE 802.15.4.

Em resumo, as principais características do dispositivo desenvolvido ao longo do trabalho e pertinentes no escopo deste projeto são:

- Alimentação via bateria de 3.7 V de íon-lítio
- Circuito de recarga da bateria via interface mini-USB
- Módulo de rádio compatível com especificação IEEE 802.15.4
- Antena monopolo com ganho de -0.5 dBi

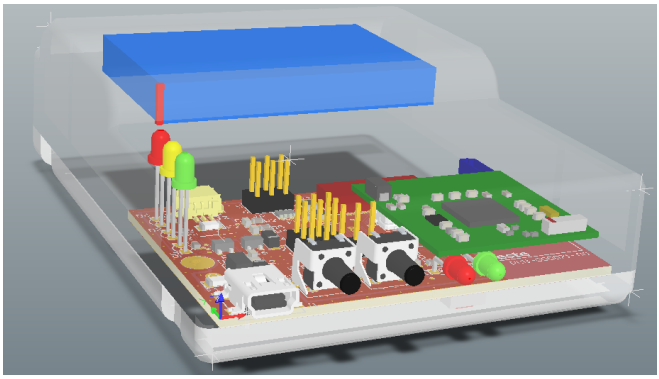


Figura 11 – Modelo tridimensional da solução completa
Fonte: elaborada pelo autor

Figura 12 – Módulo montado, com bateria
Fonte: tirada pelo autor

- Interface UART – permite *debugging* do dispositivo e interface com outro dispositivo via SLIP

- Interface Boreste 3SI para interface com dispositivos Boreste como sensores, atuadores e interfaces homem-máquina

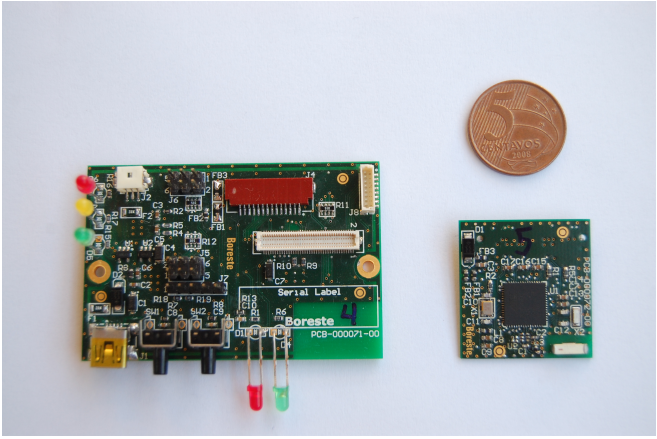


Figura 13 – Base e Módulo de rádio
Fonte: tirada pelo autor

5.3 GATEWAY

A ideia de uma rede de sensores/atuadores sem fio 6LoWPAN, apesar de permitir o uso do padrão IPv6, não tem seu potencial totalmente explorado caso a rede permaneça isolada. As vantagens de utilizar tal padrão se tornam menos significantes se não há comunicação direta com outros dispositivos IPv6 fora do *link* local. O uso de padrões fechados e mais simples novamente se torna uma opção interessante nestes casos.

A integração dos dispositivos na WPAN a uma rede IPv6 convencional é ponto chave do conceito de Internet das Coisas. Os dispositivos na rede 6LoWPAN, apesar de sua simplicidade característica, passam a ser acessíveis de qualquer tipo de rede compatível com IPv6 e da própria Internet.

Algumas diretrizes foram estabelecidas na etapa de planejamento do *gateway* no sentido de manter o dispositivo simples sem comprometer sua funcionalidade e utilidade para os testes pretendidos. Uma das principais diretrizes diz respeito à simplicidade e implementação do hardware. Decidiu-se que o *gateway* deve ser composto por um nó, já exposto anteriormente, conectado a um dispositivo com interface Ethernet, com os dados trafegando entre ambos via interface ponto-a-ponto.

5.3.1 Requisitos

Um dispositivo de *gateway* deve possuir pelo menos duas interfaces de rede, neste caso uma compatível com a WPAN IEEE 802.15.4 e outra com a rede Ethernet. A lógica que conecta essas duas interfaces deve:

1. **Fazer a ponte entre as redes**: deve realizar a conversão a nível da camada de enlace e física entre as redes IEEE 802.15.4 e Ethernet.

2. **Implementar a camada de adaptação 6LoWPAN na interface com a rede sem fio**: a camada de adaptação é responsável pela compressão e descompressão de pacotes, fragmentação e tradução de endereços.

3. **Realizar o roteamento**: os pacotes que chegam a uma interface devem ser roteados para a interface apropriada.

5.3.2 Hardware

A interface com a rede sem fio já está contemplada nos dispositivos desenvolvidos. O *gateway* deve possuir outra interface de rede, de preferência que permita comunicação direta com a Internet. A interface selecionada para implementação do Gateway foi a Ethernet, mas outras interfaces, como por exemplo Wi-Fi® ou GPRS/EDGE, poderiam ser selecionadas. A interface de rede Ethernet, porém, exige recursos de hardware que estão além do que os dispositivos em questão disponibilizam, principalmente no que diz à capacidade de processamento e também de memória RAM para alocação dos *buffers*.

Um dispositivo com maior quantidade de recursos e que possui software e hardware compatíveis com uma rede Ethernet IPv6 padrão foi escolhido. Visando reaproveitar o hardware para rede sem fio já desenvolvido, portanto, o *gateway* deve possuir a estrutura da Figura 14, que determina em alto nível as conexões entre as partes do dispositivo.

O dispositivo de Interface Ethernet IPv6 deve, no contexto deste projeto:

- Possuir interface Ethernet.

- Executar Sistema Operacional que permita programação alto nível do sistema e possua pilha TCP/IPv6 completa e validada.

- Possuir recursos de memória RAM e Flash e velocidade compatíveis.

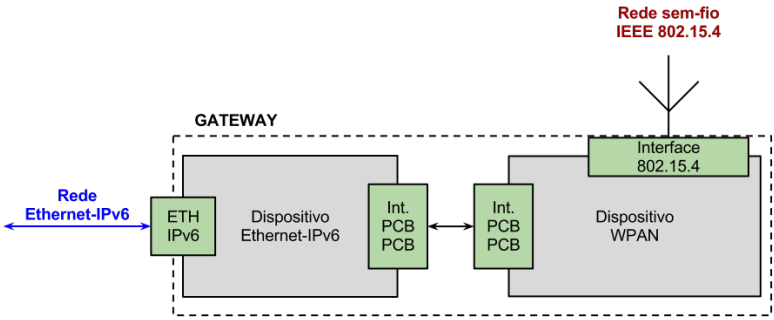


Figura 14 – Estrutura simplificada do Gateway
Fonte: elaborada pelo próprio autor

- Possuir interface para comunicação PCB-PCB com dispositivo IEEE 802.15.4 – interface a ser definida nas próximas seções.

No mundo dos sistemas embarcados de maior desempenho predomina o uso de sistemas operacionais cujo núcleo é o Linux. É vasta a quantidade de plataformas de hardware que permitem que sistemas baseados em Linux sejam portados e operem de forma confiável. O núcleo Linux possui código aberto, sendo desenvolvido por diversas empresas em conjunto com a comunidade.

O principal benefício neste projeto em usar uma plataforma Linux é o ganho em tempo de desenvolvimento. O fato de possuir uma pilha IPv6 validada e toda estrutura que um Sistema Operacional consolidado possui abrevia o tempo de trabalho e permite que parte do *gateway* seja considerada validada e operante da maneira esperada.

Várias plataformas Linux foram consideradas, principalmente sob a forma de SBC. Dada a vasta quantidade de opções no mercado as plataformas consideradas não serão aqui expostas – na próxima seção a plataforma escolhida é apresentada de forma direta.

5.3.2.1 Dispositivo de Interface com rede Ethernet

O projeto Beagleboard, desenvolvido em conjunto pela Texas Instruments[19] e pela Digikey[20] tem o objetivo de desenvolver uma placa para

[19] http://www.ti.com
[20] http://www.digikey.com

servir de demonstração dos microcontroladores de arquitetura ARM Cortex-A8 e ao mesmo tempo ser uma plataforma de aprendizado para Linux em sistemas embarcados (TEXAS INSTRUMENTS AND DIGIKEY, 2008). O projeto possui em seu portfólio quatro placas principais: a BeagleBoard, a BeagleBoard-xM, a BeagleBone e a BeagleBone Black, a última tendo sido lançada durante o desenvolvimento deste trabalho. A BeagleBone foi selecionada para este projeto pois:

- Várias distribuições Linux são suportadas: Angstrom[21], Ubuntu[22] e até mesmo Android[23].
- Possui baixo custo: US$89.
- Conta com Interfaces Ethernet, serial, USB e expansão via barra de pinos de sinais do microcontrolador.
- Há vasto suporte pela comunidade.
- O hardware é aberto.

A placa e seus principais componentes são mostrados na Figura 15.

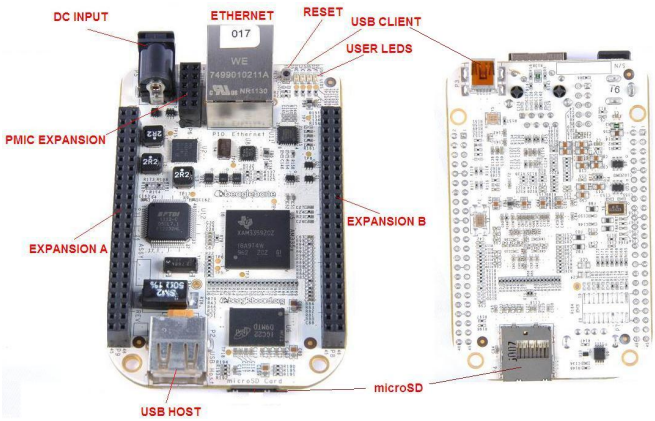


Figura 15 – BeagleBone
Fonte: (TEXAS INSTRUMENTS AND DIGIKEY, 2008)

[21] http://www.angstrom-distribution.org
[22] http://www.ubuntu.com
[23] www.android.com

5.3.2.2 Interface PCB-PCB

A comunicação entre o módulo IEEE 802.15.4 e a BeagleBone deve ser realizada via interface PCB-PCB padrão, disponível em ambos os módulos e que atenda aos seguintes requisitos:

- Velocidade que permita a transferência de dados entre ambos com taxa de transferência desejável superior a maior taxa especificada para redes IEEE 802.15.4 (250 kbps).
- Não exija conversor adicional de tensão ou protocolo.
- Tenha suporte pelo Linux e pelo Contiki.

As interfaces mais comuns neste tipo de aplicação são realizadas via os protocolos PPP e SLIP. Ambos podem funcionar sobre uma interface serial simples. O protocolo PPP possui muito mais recursos e substituiu o SLIP em muitos cenários, porém em sistemas embarcados simples o SLIP ainda é bastante utilizado devido à sua baixa complexidade e menor demanda de recursos em relação ao protocolo PPP. Dada a natureza simples dos dispositivos da WPAN em questão, o protocolo SLIP foi selecionado.

O protocolo SLIP é um padrão *de facto*, descrito na RFC1055 (ROMKEY, 1988). Em termos de hardware uma interface serial simples a dois fios mais referência é exigida. De acordo com a especificação, velocidades entre 1200 bps e 19.2 kbps são comumente utilizadas, porém velocidades superiores podem ser usadas. Ambos os lados da comunicação neste projeto suportam velocidades superiores à máxima taxa de transmissão da rede sem fio de 250 kbps, apesar da comunicação entre ambos não ter sido testada no ato desta decisão e problemas terem sido encontrados ao longo do testes, conforme descrito posteriormente.

5.3.3 Software

5.3.3.1 Distribuição Linux utilizada

A distribuição padrão utilizada na BeagleBone é a Angstrom Linux. Esta distribuição foi desenvolvida com foco em dispositivos embarcados, podendo ser executada em dispositivos com recursos limitados. Informações adicionais podem ser encontradas no site do projeto[24].

[24] http://www.angstrom-distribution.org

5.3.3.2 Interface SLIP

O protocolo SLIP é bastante simples. Os dois dispositivos conectados conhecem o endereço um do outro. Os dados enviados correspondem ao pacote, incluídos cabeçalhos da camada de rede do modelo OSI. Duas funções, uma de recepção e uma de transmissão, são necessárias. Originalmente, o pacote era enviado via serial e, em seu término, um caractere especial SLIP END. Foi proposto, e é opcional, o uso de um caractere SLIP END no início para melhorar a rejeição a erros de início de transmissão de pacotes. Para não haver problemas para enviar o mesmo caractere no meio dos dados, foi criada uma maneira de realizar *escaping*, em que uma combinação de caracteres específicos permite a diferenciação de determinado caractere como delimitador no protocolo ou como símbolo com significado específico para a aplicação.

A função de envio e recepção de dados, em "pseudocódigo", é:

```
END = 0300
ESC = 0333
ESC_END = 0334
ESC_ESC = 0335

slip_send(data, length)
  send(END)
  for c in data:
    if (c == END)
      send(ESC)
      send(ESC_END)
    else if (c == ESC)
       send(ESC)
       send(ESC_ESC)
    else
       send(c)
  send(END)

slip_recv(*buf)
  while (1)
    c = read_char()
    i = 0
    if (c == END)
      if (i > 0)
        return i
```

```
else if (c == ESC)
  c = read_char()
  if (c == ESC_END)
    buf[i++] = END
  else if (c == ESC_ESC)
    buf[i++] = ESC
else
  buf[i++] = c
```

5.3.3.3 Interface *tun*

A interface *tun* foi a forma encontrada para criar uma interface de rede virtual no dispositivo rodando Linux. A interface *tun* não precisa de um dispositivo de rede físico para funcionar – é criado um dispositivo no kernel que se comporta e possui todas as características de uma interface de rede comum. Esta interface, porém, ao contrário das interfaces físicas, é criada com permissões de usuário comum para acesso – ou seja, representa uma forma para que programas que executam no espaço de usuário tenham acesso aos dados da interface de rede.

No escopo deste trabalho, como o próprio nome da interface sugere, um túnel é criado entre a conexão serial e a interface de rede *tun*. Internamente na Beaglebone, todo o dado que chega via serial SLIP advindo do dispositivo de rede IEEE 802.15.4 é direcionado para a interface *tun*, e todo o dado que o sistema Linux roteia para a interface *tun* é passado para a interface Serial SLIP. Um utilitário, fornecido pelo Contiki e denominado *tunslip6* realiza a troca de dados entre a interface serial SLIP e a interface *tun*, realizando o tratamento necessário entre os protocolos.

As regras de roteamento e demais configurações de interfaces de rede regulares também se aplicam à interface *tun*. Criando-se as regras adequadas no dispositivo Linux, portanto, pode-se integrar a rede Ethernet IPv6 à WPAN via interface *tun* e serial SLIP. O esquema mostrando a estrutura do *gateway* utilizando esta configuração pode ser visto na Figura 16.

O programa sendo executado no nó IEEE 802.15.4 do *gateway* é bastante simples e denominado *border-router*. Esta aplicação já estava presente no Sistema Operacional e precisou ser levemente alterada em termos de tamanho de *buffers* e endereçamento no contexto deste trabalho. Na aplicação, os dados recebidos via SLIP são comprimidos e fragmentados de acordo com o padrão 6LoWPAN, sendo em seguida passados para os nós da WPAN. Dados advindos da WPAN são descomprimidos e passados utilizando

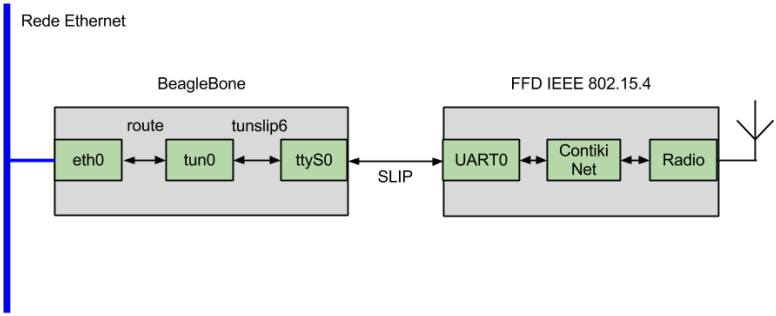


Figura 16 – Estrutura de Software do Gateway
Fonte: elaborada pelo autor

protocolo SLIP para a interface serial. O nó desenvolvido é enxergado pelo sistema Linux como uma interface de rede simples. A interface SLIP é configurada como interface de *fallback* na aplicação, ou seja, todo pacote que não possui um destino na WPAN a qual o dispositivo faz parte é encaminhado para esta interface.

A interface *tun* funciona sobre a camada de rede do modelo OSI, ou seja, os pacotes trocados entre a interface serial SLIP e a interface *tun* são pacotes IP (IPv6 no caso deste trabalho). O caminho percorrido pelos dados nas camadas do modelo OSI, considerando uma transmissão de um pacote de um dispositivo na rede sem fio para um dispositivo na rede Ethernet, pode ser visualizado no esquema da Figura 17.

Em verde na Figura 17 as camadas pelos quais os dados passam no roteamento entre dados no PC e um nó na WPAN formada. Em amarelo a camada de adaptação 6LoWPAN que realiza a compressão/descompressão dos dados, adaptação dos endereços e a fragmentação a nível de camada de enlace necessária para que os pacotes IPv6 possam ser enviados nos quadros de 127 bytes da camada MAC especificada na IEEE 802.15.4.

As linhas pontilhadas vermelhas mostram a camada na qual os dados são efetivamente passados de um dispositivo para o outro. No caso do *gateway* as camadas de transporte e aplicação não são utilizadas caso o dado não seja proveniente do *gateway* ou destinado a ele. A pilha de rede na BeagleBone não toma conhecimento da existência do dispositivo ou mesmo da rede IEEE 802.15.4. Os dados disponibilizados na interface *tun* e o encapsulamento dos mesmos para envio através dela são tratados de forma idêntica à troca de dados em qualquer outra interface de rede do dispositivo

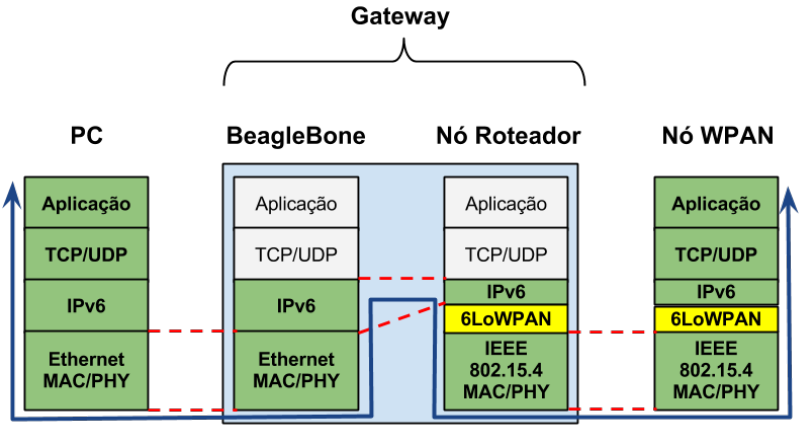


Figura 17 – Relação entre camadas do modelo OSI e *gateway* desenvolvido
Fonte: elaborada pelo autor

que suporte protocolos TCP/IP.

Em termos de software foi adicionado um script de inicialização na BeagleBone para que as interfaces sejam configuradas e os aplicativos iniciados automaticamente na inicialização do dispositivo. Foi criado um *Shell script* em formato padrão no diretório /etc/init.d denominado router6 e cujas principais linhas de código, referentes ao comando de inicialização (*start*) do script, são expostas no quadro a seguir:

```
TUNSLIP6=/usr/bin/tunslip6
TUNSLIP6ARGS="aaaa::1/64 -s /dev/ttyO1 -B 115200 -v6"
PIDFILE=/var/run/tunslip6.pid
DESC="IPv6 tunslip interface"

set -e

do_start() {
echo 20 > /sys/kernel/debug/omap_mux/uart1_rxd
echo 0 > /sys/kernel/debug/omap_mux/uart1_txd
ip -6 addr add fd13::2/64 dev eth0
echo 1 > /proc/sys/net/ipv6/conf/all/forwarding
start-stop-daemon --start --quiet --exec $TUNSLIP6
-- $TUNSLIP6ARGS
}
```

A função *do_start()* executa a seguinte sequência de comandos:

1. Configura os pinos de RX e TX da BeagleBone para que sejam utilizados com a função de UART.

2. Configura uma interface IPv6 padrão, cujo nome é eth0. A máscara fd13:: foi escolhida arbitrariamente neste projeto – o IP da interface Ethernet da BeagleBone é fd13::2 e da interface configurada no PC é fd13::1.

3. Habilita a função de roteamento.

4. Inicia o programa fornecido pelo Contiki e descrito anteriormente denominado *tunslip6*, que é responsável pelo envio e recebimento de dados via interface virtual *tun*.

Como pode ser notado no código, a interface serial é iniciada pelo *script* com velocidade de 115200 bps. A especificação do projeto citava que as interfaces ao longo do trajeto dos dados através do *gateway* deveriam possuir velocidade igual ou superior a 250 kbps, velocidade máxima de comunicação especificada na IEEE 802.15.4. Testes com velocidade superior a 115200 bps com o *setup* estabelecido, porém, falharam. O motivo é a dessincronização de relógio entre o microcontrolador do nó desenvolvido e o controlador da Beaglebone.

A comunicação foi estabelecida, porém com erros excessivos de comunicação. Várias tentativas foram realizadas utilizando diferentes velocidades de comunicação e em alguns momentos a comunicação foi possível, porém com ocorrência de erros esporádicos, tornando o dispositivo não confiável.

A configuração da interface UART no ATmega128RFA1 possui erro de 8.5% quando a velocidade de 230400 bps é selecionada, considerando frequência de operação de 16MHz. Este valor é inaceitável para comunicação assíncrona. Tentou-se utilizar a velocidade de 250000 bps, mas neste caso, como não é uma velocidade padrão, há problemas no suporte do Linux. Velocidades ainda maiores foram testadas, porém sem sucesso. Optou-se em manter a velocidade padrão de 115200 bps na qual bom funcionamento foi constatado, apesar da redução na performance no dispositivo.

Além disso, considerando os mecanismos de CSMA/CA e o desempenho do Contiki citado anteriormente, a velocidade real máxima obtida na rede sem fio isolada permaneceu em torno de 180 kbps. Tal problema não afetou de forma significativa as análises previstas no projeto e também houve pouca influência sobre os resultados esperados do trabalho.

Uma das perspectivas para o futuro é o desenvolvimento de um *gateway* integrado, com todas as funções reunidas em apenas uma placa, e utilizando uma interface síncrona mais confiável, como por exemplo SPI. O uso de controle de fluxo na serial pode também ser suficiente para a resolução do problema apontado. Tais mudanças, porém, por não serem foco deste trabalho não fazem parte do seu escopo final.

5.3.4 Implementação

A Beaglebone foi ligada ao módulo via conexão serial a 3 fios, utilizando-se os *headers* de expansão conforme a foto da Figura 18.

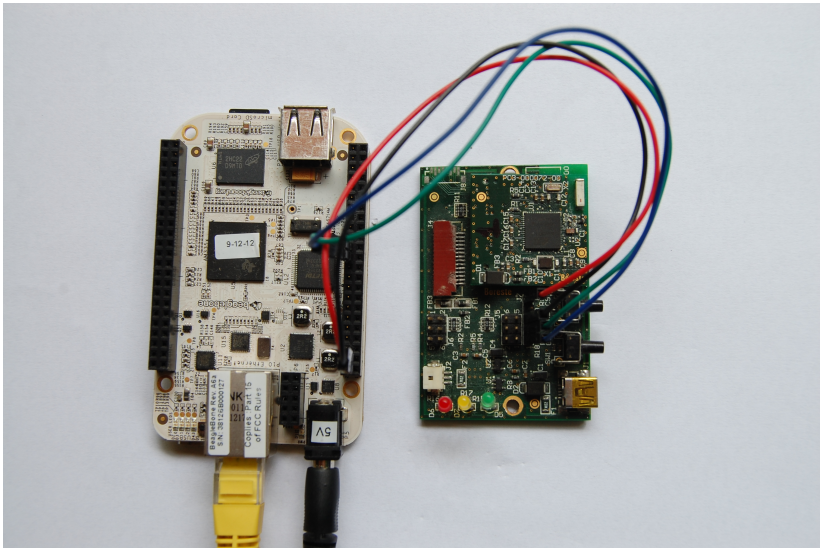


Figura 18 – *Gateway* desenvolvido
Fonte: elaborada pelo autor

O *gateway* foi alimentado com uma fonte padrão de 5V via conector da própria BeagleBone. A alimentação é passada para a base do módulo via dois dos fios exibidos na Figura 18, dispensando o uso de bateria neste módulo.

5.4 CONSIDERAÇÕES FINAIS

Os dispositivos foram projetados, implementados e construídos com sucesso. Testes simples comprovaram seu funcionamento e abriram portas para a realização de experimentos mais aprofundados envolvendo a análise do desempenho dos dispositivos e das características do uso do 6LoWPAN. Apesar de serem apenas protótipos, os dispositivos apresentam potencial para uso em aplicações comerciais dado o tamanho reduzido dos módulos e a flexibilidade de configuração proporcionada pelo uso do Linux no *gateway*.

6 ANÁLISE DOS RESULTADOS E AVALIAÇÃO DE DESEMPENHO

6.1 INTRODUÇÃO

Este capítulo apresenta a análise de resultados obtidos e promove uma avaliação de desempenho dos nodos e do *gateway* desenvolvido. Também é avaliado o comportamento do 6LoWPAN principalmente no que concerne ao desempenho dos seus mecanismos de compactação e fragmentação de datagramas. Os experimentos foram divididos em testes da rede sem fio isoladas e testes do *gateway*.

6.2 AVALIAÇÃO DA REDE SEM FIO

6.2.1 Experimento 1: avaliação da distância na taxa de perda de pacotes

O principal objetivo deste experimento é mensurar a quantidade de pacotes perdidos em função da distância entre dois nós em rede exclusiva.

Para esse experimento, assumiu-se que dois nós se comunicavam com pacotes UDP de tamanho fixo. Um nó somente enviava dados de aplicação, enquanto outro somente recebia. O nó transmissor possuía um botão que disparava o início de um teste ao ser pressionado. Em cada teste o nó transmissor enviava 5 conjuntos de 1000 pacotes de 20 bytes, a uma frequência de 50 Hz. O próprio pacote enviado continha um número de identificação do teste, que era incrementado a cada novo teste. A média da quantidade de pacotes recebidos foi calculada para cada distância.

O nó receptor permaneceu em estado de escuta. Quando recebia um pacote verificava o número de identificação do teste e mantinha um vetor com a contagem de quantos pacotes de cada teste foram recebidos. Estas estatísticas permaneceram em memória RAM e foram coletadas via console serial.

Foram realizados testes com distâncias de 15, 30, 45 e 60 metros entre os nós. Em cada intervalo de distância 5 testes eram realizados, considerando os nós com visada direta. O nó receptor calculava a média do RSSI após a recepção de cada pacote do mesmo teste. Tal valor foi armazenado junto dos resultados, também vinculado ao identificador do teste.

A configuração do Contiki utilizada corresponde à exposta na Figura 9. O rádio permaneceu ligado, sem estratégias de economia de energia,

visando não influenciar o resultado do teste.

Os resultados dos testes podem ser verificados na Figura 19. As barras no gráfico apresentam a razão entre a quantidade de pacotes recebidos e enviados em função da distância, e a linha indica o valor médio de RSSI nos testes em tal distância. Na distância de 60 metros nenhum pacote foi recebido e o valor foi omitido do gráfico para melhor visualização dos resultados.

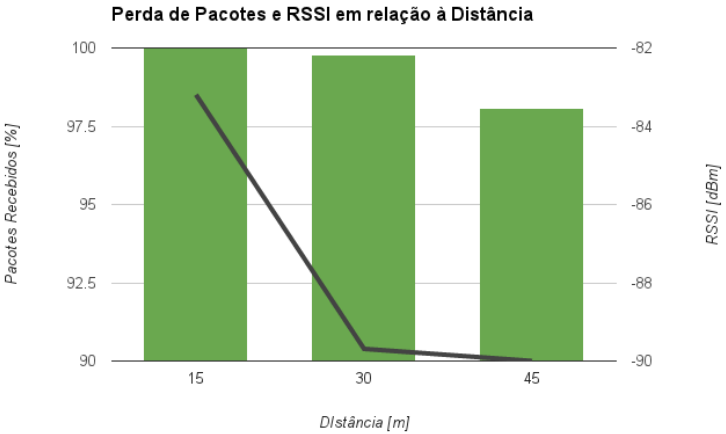


Figura 19 – Resultado do teste Distância vs PER
Fonte: elaborada pelo autor

WPANs em geral, como a própria denominação sugere, têm características de alcance curto e dispositivos simples. Redes baseadas na especificação IEEE 802.15.4 possuem alcance que varia de 10 a 100 metros em condições normais de visada direta. Nos testes obteve-se bons resultados considerando o uso de antena cerâmica SMT de pequenas dimensões e de ganho -0.5 dBi. Em distâncias de até 45 metros houve baixa perda de pacotes. A partir dos 50 metros a potência do sinal foi insuficiente para comunicação. No próprio teste com distância de 45 metros o nível de sinal estava tão baixo que o indicador de RSSI do dispositivo apontou seu limite inferior, ou seja, o mínimo valor mensurável pelo microcontrolador.

Notou-se que obstáculos, diferentes formas de posicionar os módulos e a presença de algum indivíduo em contato com a placa resultaram em grande influência nos resultados dos testes. Qualquer alteração no posicionamento de um dos nós ou a inserção de elementos próximos à rede alterava drasticamente os resultados obtidos.

Certamente o uso de antenas com maior ganho e possivelmente circuitos para amplificar a potência do sinal podem aumentar essas distâncias. Tais alterações podem ser realizadas desde que as normas sejam atendidas. No caso do projeto de irrigação mencionado na motivação desse trabalho, por exemplo, tais mudanças devem ser aplicadas caso os módulos desenvolvidos venham a ser utilizados para que o alcance ultrapasse os 150 metros.

6.2.2 Experimento 2: avaliação do consumo energético dos nós

O principal objetivo desste experimento é o de determinar o consumo dos nós desenvolvidos em três situações: transmitindo dados, recebendo dados e ocioso.

Os testes utilizaram os mesmos *firmwares* desenvolvidos no Experimento 1 para os nós transmissor e receptor em testes de medição de consumo transmitindo e recebendo dados. Para o teste de consumo de nó ocioso foi utilizado *firmware* simples exclusivo, que se mantém em *loop* sem utilizar as funções de rádio.

Todos os LEDs dos nós foram desligados, assim como demais periféricos não úteis no teste, para não influenciar as medidas. O microcontrolador foi mantido com frequência de operação de 16 MHz.

Foram realizados dois testes com *setups* distintos: utilizando um multímetro para medição da corrente e utilizando um osciloscópio para medir a diferença de potencial em um resistor de 22 Ohms em série com a bateria de íon-lítio de 3.7 V. A configuração do Contiki utilizada corresponde à exposta na Figura 9.

Os resultados obtidos podem ser resumidos da seguinte forma:

- Nó ocioso: corrente média de 10 mA com tensão de 3.89 V da bateria, resultando em potência média de 38.9 mW.
- Nó transmitindo: corrente média de 22 mA, com tensão de 3.88 V da bateria, resultando em potência média de 85.36 mW.
- Nó recebendo: corrente média de 21 mA, com tensão de 3.89 V da bateria, resultando em potência média de 81.69 mW.

Na análise da forma de onda no osciloscópio notou-se que o consumo permanecia constante quando o rádio estava ligado, em estado de transmissão ou recepção. Dada a frequência de envio de 50 Hz estabelecida, e os pacotes de aplicação de 20 bytes, o nó deveria possuir características de consumo intermitente, possuindo picos durante a transmissão. Isto caso o sistema desligasse o rádio enquanto não há envio de dados ou o mantivesse em estado

de escuta. Como mencionado, porém, o consumo permanece constante. A justificativa é a natureza da configuração do Contiki selecionada, em que não é utilizado procotolo de RDC e o rádio permanece ligado durante todo o período.

O consumo obtido é alto para uma aplicação em que o nó deva ser operado exclusivamente a bateria. Considerando a bateria de íon de lítio de capacidade de 1050 mAh utilizada o dispositivo permaneceria ligado por no máximo 48 horas. O uso de estratégias de RDC e de camadas MAC distintas é desejado nestes casos para diminuir o consumo do dispositivo (DUNKELS, 2011). O uso de tais técnicas porém aumenta a latência na comunicação. Várias tentativas foram realizadas para verificação do consumo utilizando estratégia de *Radio Duty Cycling* com o ContikiMAC, porém sem sucesso. A versão do Contiki utilizada foi a 2.6, estável, e de acordo com algumas discussões encontradas na Internet o ContikiMAC ainda não está funcional no ATmega128RFA1. O CX-MAC também não funcionou.

O consumo do microcontrolador, de acordo com o *datasheet* do dispositivo (ATMEL, 2013), é de 4.1 mA quando somente CPU está ligada (sem periféricos não essenciais) a 16 MHz, 14.5 mA quando em recepção e 16.5 mA durante transmissão. Os resultados dos testes estão condizentes com tais valores – as diferenças de consumo são causadas principalmente pelo consumo dos demais componentes da placa e periféricos sendo utilizados no microcontrolador, como o próprio gerenciador das entradas/saídas.

6.2.3 Experimento 3: análise da compressão de dados IPHC/NHC no Contiki

O objetivo desse experimento é o de verificar a compressão IPHC/NHC implementada no Sistema Operacional adotado na prática, considerando troca de dados na rede desenvolvida. Almeja-se analisar bit a bit os dados antes e depois da compressão IPHC/NHC.

Para esse experimento, um nó foi programado para enviar um pacote UDP simples contendo um comando de três bytes que determina a cor do LED a ser ligado no dispositivo remoto. Na análise exposta no documento foi utilizado um comando para ligar o LED vermelho. Para tal estabeleceu-se que a palavra *red* deveria ser enviada. Foi inserido código no Sistema Operacional para imprimir via console serial o pacote antes e depois da compressão na camada de adaptação 6LoWPAN do transmissor. Em seguida uma análise bit a bit foi realizada e comparada à especificação RFC6282 (HUI; THUBERT, 2011). A comunicação utilizou o intervalo de portas UDP passíveis de compressão a um byte (para as duas portas), e o pacote foi enviado em *link*

local, caso no qual, de acordo com o que foi exposto nas seções anteriores, atinge-se a máxima compressão.

Inicialmente analisou-se os dados antes da compressão. Os campos do cabeçalho IPv6, totalizando 40 bytes, são interpretados na Tabela 4. Os campos do cabeçalho UDP, totalizando 8 bytes, são analisados na Tabela 5.

Tabela 4 – Análise cabeçalho IPv6, sem compressão, *unicast* local

Campo	Valor	Significado
Versão	6	IPv6
Classe do tráfego	00	Classe 0, sem diferenciação
Marcação do fluxo	00000	Tráfego Regular
Tamanho do Payload	000b	11 bytes
Tipo do próximo cabeçalho	11	UDP
Limite de saltos	40	64 nós
Endereço de origem	fe8000000000000000000fffc000002	fe80::ff:fc00:02
Endereço de destino	fe8000000000000000000fffc000001	fe80::ff:fc00:01

Tabela 5 – Análise cabeçalho UDP, sem compressão – *unicast* local

Campo	Valor	Significado
Porta de origem	f0b1	Porta 61617
Porta de destino	f0b0	Porta 61616
Tamanho Cabeçalho UDP + dados	000b	11 bytes
Checksum UDP	510b	Checksum = 0x510b

Em seguida a compressão 6LoWPAN foi analisada. A Tabela 6 apresenta o cabeçalho IP após compressão IPHC. Na Tabela 7 estão descritos os campos do cabeçalho UDP após compressão NHC.

Tabela 6 – Análise cabeçalho IP comprimido – *unicast* local

Campo	Valor	Significado
Dispatch Byte	7e	Identifica a compressão IPHC, especifica que a classe de tráfego e marcação de fluxo são omitidas, o próximo cabeçalho é codificado utilizando NHC, e o valor de Hop Limit é 64
IPHC Header	33	Sem identificador de contexto adicional, endereços de origem e destino usam compressão sem estado, endereços de origem e destino completamente omitidos, endereço de destino não é *multicast*

Tabela 7 – Análise cabeçalho UDP comprimido – *unicast* local

Campo	Valor	Significado
NHC Dispatch byte	f3	NHC UDP Dispatch Byte, 16 bits de checksum na mensagem, portas UDP comprimidas
Portas UDP Comprimidas	10	Porta de origem = 0xf0b1, porta de destino = 0xf0b0
UDP Checksum	510b	Checksum = 0x510b

A compressão empregada na comunicação *unicast* local implementada no Contiki correspondeu ao especificado na RFC6282 (HUI; THUBERT, 2011). O uso de portas com prefixo 0xf0b (intervalo de portas entre 61616 e 61631) permitiu compressão adicional de 3 bytes no cabeçalho UDP – tal compressão porém vem com um preço: pelo número de portas limitado que permite compressão, é provável a presença de conflitos entre aplicações distintas com a mesma porta utilizada.

A compatibilidade com a especificação é importante para casos de interoperabilidade com nós executando outros sistemas operacionais, como o TinyOS, por exemplo. A compatibilidade entre esses dois sistemas operacionais é abordada na literatura, como em (KO et al., 2011a) e (KO et al., 2011b). Os testes aqui presentes são simples e não visam atestar a compatibilidade, tarefa merecedora de um trabalho próprio. O objetivo aqui é verificar as particularidades da comunicação e a compatibilidade em determinados casos. A compatilidade total só poderia ser ceritificada caso todos os casos de comunicação possíveis fossem analisados.

6.2.4 Experimento 4: compressão de dados e velocidade – comparação entre LoWPAN_IPHC e IPv6 sem compressão

O principal objetivo deste experimento, considerando que o 6LoWPAN foi concebido para reduzir a quantidade de *overhead* na transmissão de dados entre nós da rede IEEE 802.15.4 utilizando IPv6, consiste em verificar o impacto em termos de velocidade e de quantidade de dados úteis transmitidos com e sem compressão. A compressão utilizada segue a RFC6282.

Com este objetivo foram programados dois nós, um cliente e um servidor, se comunicando via UDP. Os nós foram posicionados a 3 metros de distância um do outro, com RSSI médio de -76 dBm no nó receptor. O nó cliente enviava um pacote ao servidor, que respondia imediatamente com os mesmos dados de aplicação. O nó cliente enviava outro pacote assim que recebia a resposta do primeiro. Caso houvesse *timeout*, o aplicativo gravava a estatística e o teste era considerado inválido.

O teste foi realizado com diferentes tamanhos de pacotes, variando de 20 bytes de carga útil a 520 bytes. A carga de cada camada foi analisada: MAC, rede, transporte e camada de aplicação (carga útil). Foi feita a comparação do impacto dos dados inseridos em cada camada no total da troca de dados, considerando a soma de recepção e transmissão. Os resultados dos testes com compressão IPHC/NHC e sem compressão foram comparados ao final do teste.

O termo sem compressão foi utilizado para referenciar o pacote

IPv6 sendo enviado com cabeçalho completo. Assume-se, porém, que a fragmentação da especificação do 6LoWPAN continua ocorrendo, caso contrário pacotes com dados de aplicação de tamanhos superiores a 56 bytes não poderiam ser enviados, levadas em consideração as configurações do teste. A mudança entre o teste com e sem compressão reside apenas na forma como os dados de cada camada são enviados em seus respectivos cabeçalhos, comprimidos ou não.

A Figura 20 exibe a quantidade de dados enviados por segundo, separados por camada, sem utilizar compressão, ou seja, considerando IPv6 puro. O teste foi realizado variando-se o tamanho dos pacotes de aplicação. A Figura 21 mostra os resultados dos mesmos testes, porém utilizando compressão IPHC/NHC. A Figura 22 mostra a quantidade de bytes por camada em cada pacote, considerado individualmente, sem compressão.

A Figura 23 também considera apenas pacotes individuais com tamanho variando no eixo das abscissas e a quantidade de dados por camadas com compressão IPHC/NHC. Fica claro nas figuras 22 e 23 os tamanhos de pacote em que é necessário o uso de um novo fragmento (incremento na quantidade de dados na camada MAC). A razão entre carga útil e carga total é mostrada no gráfico da Figura 24 para a comunicação com e sem compressão.

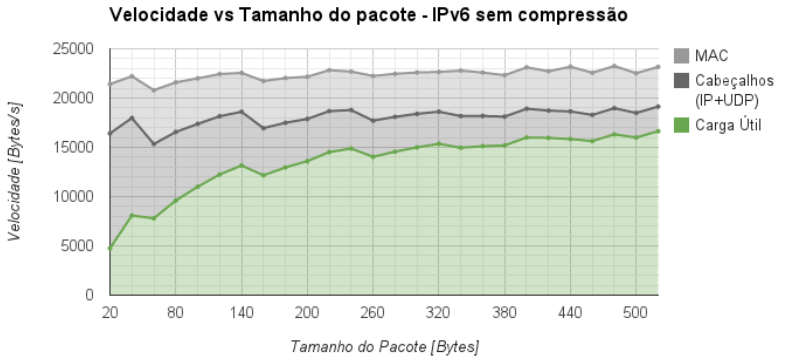


Figura 20 – Quantidade de dados por camada – IPv6
Fonte: elaborada pelo autor

Em termos de quantidade total de bytes os resultados são iguais em ambos os casos, como era de se esperar. A carga útil enviada quando utilizando compressão supera a quantidade de dados úteis enviados no caso sem compressão em um mesmo intervalo de tempo. O *overhead* dos cabeçalhos na ausência de compressão se torna menos significativo a medida

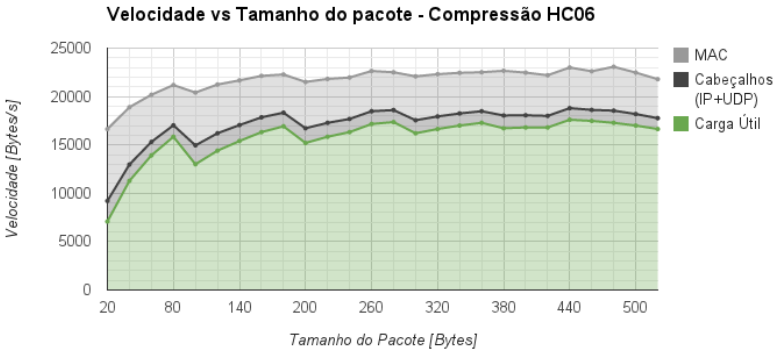


Figura 21 – Quantidade de dados por camada – Compressão HC06
Fonte: elaborada pelo autor

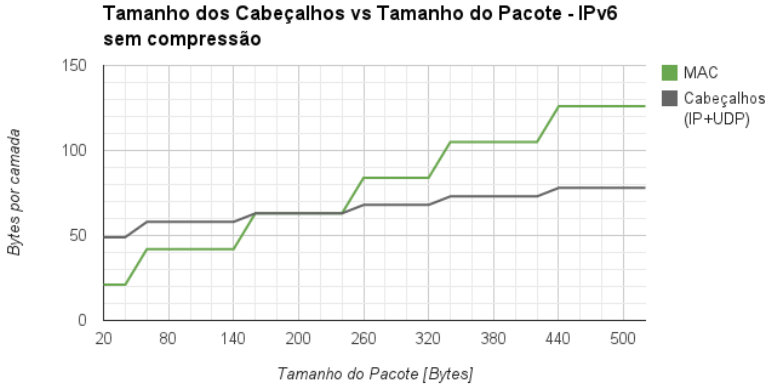


Figura 22 – Quantidade de bytes das camadas MAC, IP e UDP em cada transmissão de pacote – IPv6 sem compressão
Fonte: elaborada pelo autor

que o tamanho do pacote aumenta – isso porque a compressão acontece no cabeçalho principal de determinado pacote, que só é enviado no primeiro fragmento no caso em que a fragmentação é necessária. Demais fragmentos possuem cabeçalho de tamanho constante, são pequenos e idênticos em ambos os casos (com ou sem compressão).

É claro nos resultados o ganho em termos de carga útil enviada ao

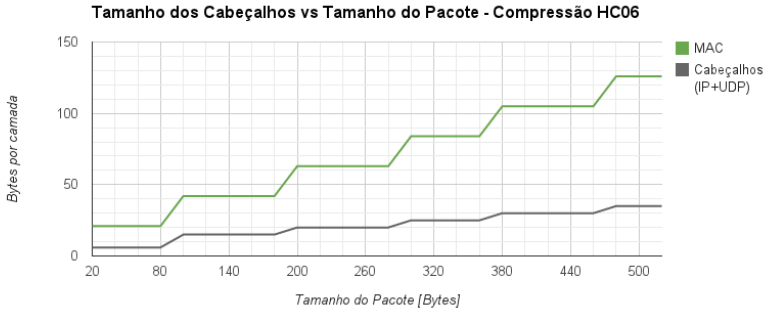


Figura 23 – Quantidade de bytes das camadas MAC, IP e UDP em cada transmissão de pacote – compressão IPv6

Fonte: elaborada pelo autor

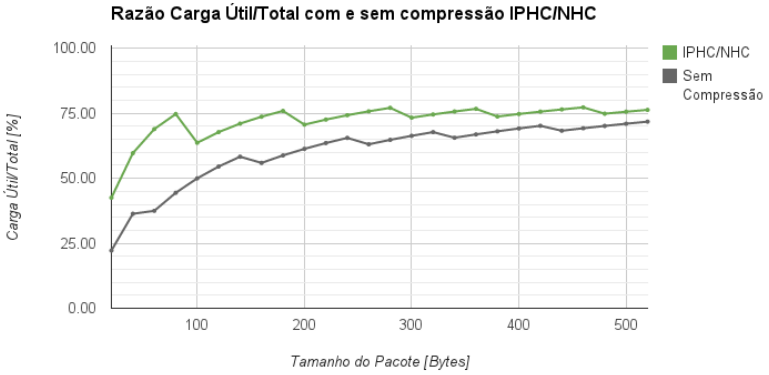


Figura 24 – Razão entre carga útil e carga total com e sem compressão IPHC IPHC

Fonte: elaborada pelo autor

se utilizar a compressão de cabeçalhos. O gráfico da razão carga útil por carga total corrobora a motivação para uso de redes 6LoWPAN sobre IEEE 802.15.4: redes de baixa velocidade com troca de pacotes de poucos bytes de carga útil têm desempenho melhorado significativamente ao se utilizarem técnicas de compressão.

Na transmissão de pacotes de 80 bytes, por exemplo, a velocidade

de envio de carga útil com compressão é de aproximadamente 15 KiB/s, enquanto a velocidade sem compressão é de cerca de 9.5 KiB/s. Outro ponto que evidencia o melhor desempenho utilizando a compressão diz respeito à fragmentação. A quantidade limite de carga útil, em que o pacote passa a precisar de fragmentação na subcamada MAC, é substancialmente diferente no caso em que há compressão. Isso porque o cabeçalho passa a ser muito menor, tornando esse limite maior quanto maior a compressão.

Nos gráficos das figuras 22 e 23 os saltos na quantidade de bytes de cabeçalho da camada MAC correspondem ao aumento de uma unidade no número de fragmentos enviados. Pode-se notar que no caso em que a compressão é utilizada os incrementos na quantidade de fragmentos são realizados com tamanhos maiores de pacotes. Esta diferença de limiar pode causar diferenças notáveis de performance em certos cenários. Se o tamanho do datagrama está próximo do limite para a necessidade de um novo fragmento o uso da compressão pode evitar que dois fragmentos sejam necessários, enviando todos os dados em apenas um quadro na subcamada de enlace.

A IEEE 802.15.4 especifica taxas de transmissão de até 250 kbps. Nos testes a taxa de transmissão, considerando a soma entre dados enviados e transmitidos, chegou a cerca de 180 kbps, resultado semelhante ao obtido por (FAROOQ; KUNZ, 2012), que também utiliza o Contiki e justifica a menor taxa de transferência na saturação do canal. A taxa de transmissão de 250 kbps também não considera o uso do CSMA/CA, adotado nos testes. A própria dinâmica do Sistema Operacional Contiki, tal como a implementação da pilha uIPv6, o gerenciamento de tarefas, entre outros fatores, também influenciam no resultado final.

Outro fenômeno que se pode notar nos gráficos é que quanto maior a quantidade de dados por fragmento maior a taxa de transmissão de bytes no total. Tal valor tende a se estabilizar em 180 kbps. Pacotes menores implicam em maior chaveamento entre transmissão e recepção nos testes, maior quantidade de conflitos no algoritmo CSMA/CA, e maior influência das particularidades do Sistema Operacional nos resultados, justificando tal observação.

6.3 AVALIAÇÃO DO GATEWAY

Os testes nessa seção compreendem a validação do *gateway* como solução. Uma conexão Ethernet ponto-a-ponto simples entre um PC rodando Linux e o *gateway* foi utilizada nos testes. A Figura 25 exibe a topologia da rede básica construída para os testes.

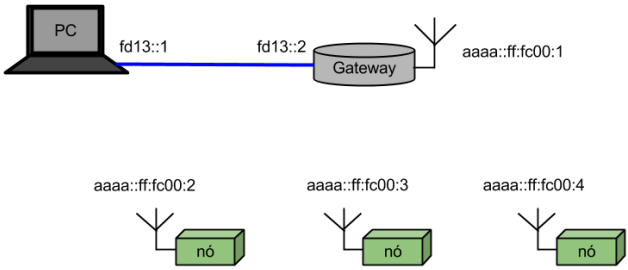


Figura 25 – Topologia utilizada nos testes do *gateway*
Fonte: elaborada pelo autor

6.3.1 Experimento 5: RTT – *Round Trip Time*

O objetivo deste experimento foi o de determinar o RTT entre o PC e um nó na WPAN. Um programa simples sendo executado no PC, implementado em linguagem Python, enviava uma sequência de pacotes UDP para o nó na WPAN e calculava a média de tempo de resposta. Apenas um nó foi utilizado na rede sem fio (além do *gateway*). O nó possuía *firmware* simples que escutava pacotes UDP e ecoava qualquer pacote recebido de volta para a fonte. A média de tempo entre envio e recepção da resposta foi calculada no PC. O tempo de processamento dos dados pelo PC é rápido o suficiente sendo sua influência desconsiderada no resultado do teste.

Além disto, com o mesmo *setup* montado, foi utilizada a ferramenta *ping6* para determinar o tempo que cada dispositivo no caminho até o nó leva para responder. Medidas realizadas incluem a Beaglebone, o nó sem fio do *gateway* e o próprio nó na rede IEEE 802.15.4. Tal feito permite calcular o tempo aproximado que o pacote leva percorrendo cada trecho no caminho entre o PC e o nó na WPAN.

O gráfico da Figura 26 mostra os resultados obtidos nos testes com o programa de eco de pacotes UDP.

Nos testes com a ferramenta *ping6* foi considerada a média de 20 *pings* para cada IP no caminho para o nó na WPAN. Os parâmetros utilizados na chamada do programa *ping6* foram os padrões: *payload* de 56 bytes somados aos 8 bytes do cabeçalho ICMP, totalizando 64 bytes de dados na mensagem enviada. Os resultados estão descritos na Tabela 8.

De posse de tais dados pode-se estimar o tempo que cada pacote de *ping* leva entre cada interface. O tempo entre a interface Ethernet do PC e da interface Ethernet da Beaglebone é obtido diretamente a partir do *ping*.

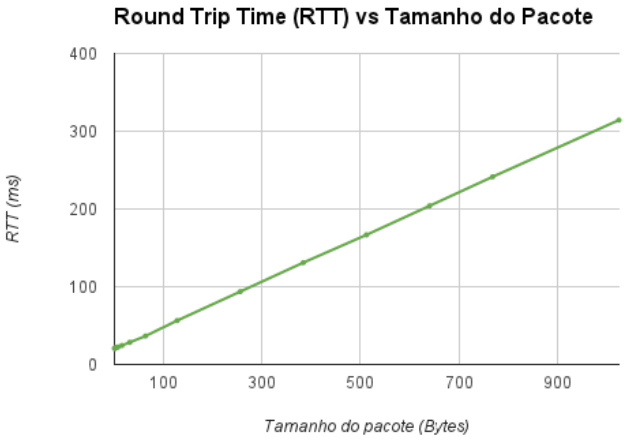


Figura 26 – RTT comunicação PC com nó na rede sem fio
Fonte: elaborada pelo autor

Tabela 8 – Média da resposta do comando Ping de cada dispositivo no caminho para nó na WPAN

Dispositivo	IP	**Tempo de resposta**
Beaglebone	fd13::2	0.257ms
Interface IEEE 802.15.4 do Gateway	aaaa::ff:fc00:1	24.772ms
Nó na WPAN	aaaa::ff:fc00:2	35.547ms

Para calcular o tempo que o pacote leva para ir da Beaglebone ao nó IEEE 802.15.4 do *gateway* via interface SLIP subtrai-se o valor do *ping* para o nó do valor de *ping* para a Beaglebone, dividindo por 2 para calcular o tempo no envio e na resposta, O tempo entre o nó do *gateway* e o nó na WPAN é obtido de forma análoga. Na Figura 27 é exibida a latência do pacote em cada trecho entre a origem e o nó na WPAN.

O crescimento do RTT é linear em relação ao tamanho do pacote. A velocidade da conexão serial entre Beaglebone e nó IEEE 802.15.4 é o gargalo na implementação do *gateway*. A velocidade máxima da UART entre BeagleBone e módulo de 115200 bps amplia o RTT e limita a taxa de transmissão. Apesar disso, os valores obtidos para o RTT estão de acordo com os requisitos de aplicações que não exijam taxas de amostragem altas. Considerando apenas um nó na rede, por exemplo, frequências de amostragem de 20 Hz podem ser facilmente obtidas para pacotes UDP de

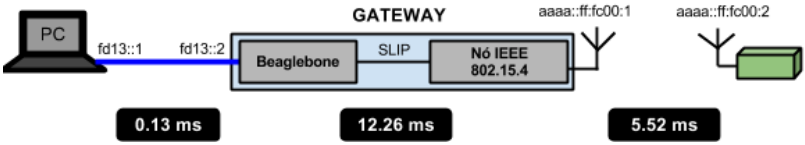


Figura 27 – Latência no caminho do pacote
Fonte: elaborada pelo autor

6 bytes, como foi constatado no Teste 8 neste mesmo capítulo.

6.3.2 Experimento 6: velocidade de transmissão de carga útil variando tamanho do pacote

O principal objetivo deste experimento é avaliar a taxa de transmissão de carga útil viável através do *gateway* implementado. A estrutura usada no teste é idêntica àquela usada no Teste 5. Ao invés de calcular RTT, a quantidade de bytes transmitidos e enviados entre PC e nó na WPAN foi verificada e a velocidade foi calculada.

O gráfico da Figura 28 ilustra os resultados obtidos neste experimento.

É possível observar-se que a velocidade de transmissão cresce a medida que o tamanho do pacote aumenta (menor custo para transmissão de cabeçalhos das camadas inferiores). A velocidade estabiliza com pacotes de tamanho superior a 1 KiB – as mesmas limitações comentadas no teste anterior se aplicam aqui, sendo o principal gargalo a velocidade da conexão serial entre Beaglebone e nó IEEE 802.15.4 no *gateway*. Tamanhos de pacote são limitados a 1220 bytes dada restrição de recursos de RAM, que limita os *buffers* de recepção e transmissão.

O fato de pacotes maiores resultarem em velocidades maiores pode influenciar o projeto de redes de sensores ou outros tipos de dispositivos em que a amostragem sequencial de valores de cada nó da rede não é necessária. Nestes casos há a possibilidade de cada nó manter em buffer interno os dados de interesse, transmitindo vetores de dados encapsulados em pacotes maiores. Equipamentos de *log* de dados, por exemplo, podem se beneficiar de tal estratégia.

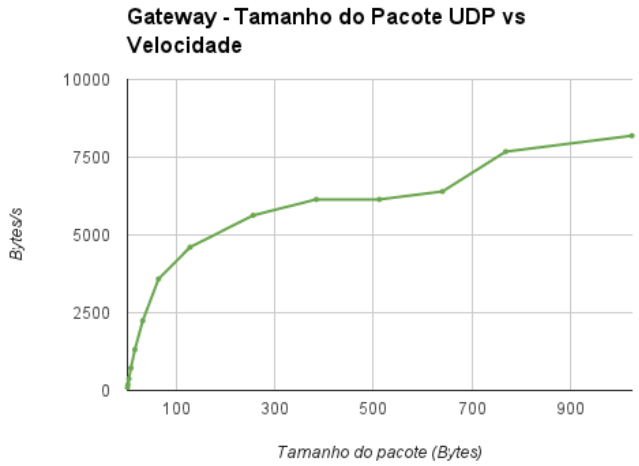


Figura 28 – Velocidade de comunicação – recepção e transmissão combinadas
Fonte: elaborada pelo autor

6.3.3 Experimento 7: verificação do pacote UDP enviado da rede WPAN para PC na rede IPv6 externa

Neste experimento tem-se como objetivo validar a compressão IPHC/NHC implementada no Contiki de pacote UDP enviado da rede sem fio para dispositivo localizado na rede externa IPv6 e comparar com a estrutura do pacote definida na RFC6282.

Foi utilizada a topologia da Figura 25, com apenas um nó na WPAN. Um nó recebia uma requisição simples do PC e respondia com valor de RSSI em um pacote UDP com carga útil de 6 bytes. O conteúdo do pacote foi impresso via console serial através de alteração no código do Contiki em parte específica do código para análise. O teste foi realizado com compressão IPHC/NHC habilitada e também com dados sem compressão. As portas UDP utilizadas se encontram no intervalo de portas passível de compressão.

O primeiro teste considerou dados sem compressão. Os campos do cabeçalho IPv6, totalizando 40 bytes, são interpretados na Tabela 9. Os campos do cabeçalho UDP, totalizando 8 bytes, são analisados na Tabela 10. O pacote analisado é o pacote com origem no nó da WPAN com destino para o PC.

Em seguida a compressão 6LoWPAN foi analisada. A Tabela 11

Tabela 9 – Análise cabeçalho IPv6, sem compressão, comunicação com dispositivo externo

Campo	Valor	Significado
Versão	6	IPv6
Classe do tráfego	00	Classe 0, sem diferenciação
Marcação do fluxo	00000	Tráfego Regular
Tamanho do Payload	000e	14 bytes
Tipo do próximo cabeçalho	11	UDP
Limite de saltos	40	64 nós
Endereço de origem	aaaa00000000000000000fffc000002	aaaa::ff:fc00:02
Endereço de destino	fd13000000000000000000000000001	fd13::2

Tabela 10 – Análise cabeçalho UDP, sem compressão, comunicação com dispositivo externo

Campo	Valor	Significado
Porta de origem	f0b0	Porta 61616
Porta de destino	f0b1	Porta 61617
Tamanho Cabeçalho UDP + dados	000e	14 bytes
Checksum UDP	965c	Checksum = 0x965c

apresenta o cabeçalho IP após compressão IPHC. Na Tabela 12 estão descritos os campos do cabeçalho UDP após compressão NHC.

Tabela 11 – Análise cabeçalho IP comprimido – *unicast* local

Campo	Valor	Significado
Dispatch Byte	7e	Identifica a compressão IPHC, especifica que a classe de tráfego e marcação de fluxo são omitidas, o próximo cabeçalho é codificado utilizando NHC, e o valor de Hop Limit é 64
IPHC Header	f0	Sem identificador de contexto adicional, endereço de origem usa compartilhamento de contexto e é completamente omitido. Endereço de destino não é *multicast*, não utiliza compartilhamento de contexto e é completamente inserido no cabeçalho.

Tabela 12 – Análise cabeçalho UDP comprimido – *unicast* local

Campo	Valor	Significado
NHC Dispatch byte	f3	NHC UDP Dispatch Byte, 16 bits de checksum presentes na mensagem, portas UDP comprimidas
Portas UDP Comprimidas	01	Porta de origem = 0xf0b0, porta de destino = 0xf0b1
UDP Checksum	965c	Checksum = 0x965c

Os resultados novamente estão de acordo com o que especifica a RFC6282. A Figura 4 estabelece que na comunicação com nós da rede externa o tamanho dos cabeçalhos pode chegar a apenas 10 bytes. É assumido na figura, porém, que há compartilhamento de contexto, o que não ocorre neste caso uma vez um dos nós não se encontra em uma rede 6LoWPAN.

O endereço de destino está, no resultado do teste, completamente contido no cabeçalho.

O endereço de origem pôde ser comprimido ao sair do nó na WPAN pois possui o mesmo prefixo da rede do nó do *gateway* e utiliza o EUI-64 na formação do endereço. O *gateway*, ao descomprimir os cabeçalhos e encaminhar o pacote para a rede Ethernet IPv6 insere no campo de origem o endereço IP formado, aaaa::ff:fc00:2. Em relação à compressão do cabeçalho UDP as mesmas conclusões do teste 3 se aplicam.

Comparando com a quantidade de dados nos cabeçalhos IPv6 e UDP sem compressão foi possível a redução de 25 bytes, com 48 bytes de cabeçalho antes e 23 bytes de cabeçalhos após a compressão.

6.3.4 Experimento 8: carregamento de página via HTTP

O objetivo neste experimento é avaliar os tempos de resposta e comparar o *overhead* com e sem compressão IPHC no carregamento de páginas via HTTP. Foi testado o uso de nós na rede sem fio como servidores de páginas simples, utilizando protocolo HTTP na comunicação. A possibilidade de utilizar tal protocolo viabiliza o uso de técnicas REST para interação com nós da WPAN, permitindo ainda maior compatibilidade com dispositivos conectados à Internet.

Um nó na rede WPAN hospedava páginas com conteúdo de tamanho configurável. O conteúdo era servido via TCP na porta 80. Requisição e resposta utilizavam protocolo HTTP. Um programa Python no computador fazia uma requisição HTTP, respondida pela placa com página simples de conteúdo fixo. Para o teste não foi implementado um servidor completo no nó, somente um servidor bastante simples que suportava apenas requisições HTTP GET na página raiz e respondia com texto puro, sem tratamento de erros, hospedagem de imagens ou outros recursos.

A página era gerada dinamicamente enquanto era enviada, sem *buffers* intermediários. Páginas com tamanho superior ao tamanho máximo do *buffer* eram geradas em partes. Cada bloco de N bytes era enviado em um pacote TCP. A mesma conexão TCP era mantida durante todo o envio da página requisitada. Páginas de 16 bytes, 100 bytes, 10 KiB e 100 KiB foram consideradas nos testes. O uso de um programa Python simples na requisição da página permitiu a medição de tempo com maior precisão e sem influência de *overhead* decorrente do uso de algum navegador.

O teste com página de 100KiB à primeira vista pode parece desproporcional ao propósito dos dispositivos sendo avaliados. Há aplicações, porém, como por exemplo sistemas com *logging*, que a conexão

com dispositivos não é permanente. O dispositivo pode passar dias coletando dados de um sensor e tê-los posteriormente requisitados de só uma vez, justificando testes com quantidades maiores de dados.

Foram analisados os dados trocados considerando uso de compressão IPHC e IPv6 puro. Foi medido o tempo necessário entre a requisição e envio da página completa ao computador para comparação.

A Figura 29 mostra a quantidade relativa de bytes por camada em relação ao total de bytes trocados no carregamento de páginas com 16 bytes e 100 KiB, considerando cenários com e sem compressão.

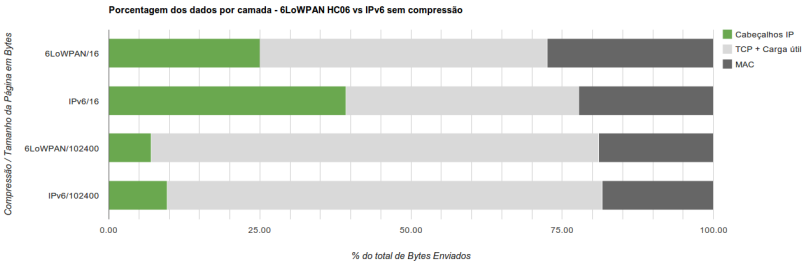


Figura 29 – Quantidade de bytes por camada em relação ao total transmitido
Fonte: elaborada pelo autor

A quantidade absoluta de dados trafegados em cada camada para requisição e resposta de uma página de 16 bytes pode ser verificada no gráfico da Figura 30. O mesmo resultado é exibido para uma página de 100 KiB na Figura 31.

O tempo médio de carregamento da página, considerados 5 testes para cada tamanho de página é exibido na Tabela 13, considerando IPv6 sem compressão. Os tempos para os mesmos testes, utilizando compressão IPHC, são mostrados na Tabela 14.

Tabela 13 – Análise tempos de carregamento das páginas – IPv6 sem compressão

Tamanho da página	Tempo
16 bytes	115 ms
100 bytes	128 ms
10 KiB	1820 ms
100 KiB	22080 ms

A análise dos dados trocados entre cliente e servidor no experimento

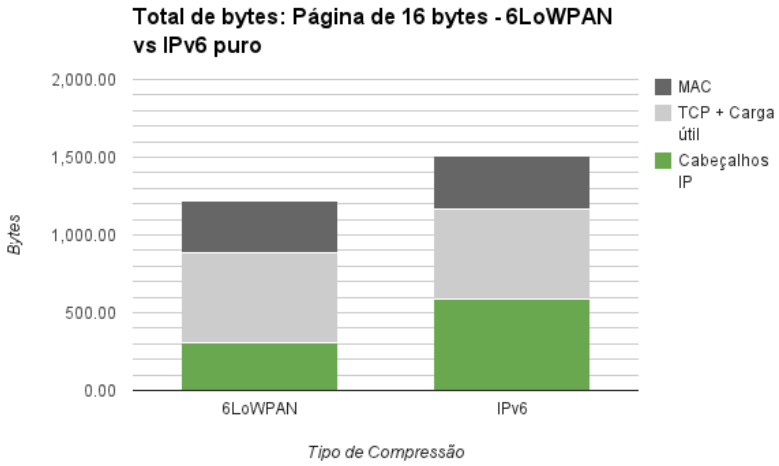


Figura 30 – Quantidade absoluta de bytes enviados pelo servidor. Página de 16 bytes

Fonte: elaborada pelo autor

Tabela 14 – Análise dos tempos de carregamento das páginas – utilizando compressão IPHC

Tamanho da página	Tempo
16 bytes	111 ms
100 bytes	123 ms
10 KiB	1810 ms
100 KiB	21750 ms

com a página de 16 bytes, realizada utilizando a aplicação Wireshark [1], pode ser vista na Figura 32.

Como conclusão, pode-se observar que páginas pequenas implicam em carga útil pequena o que resulta em maior efetividade da compressão no 6LoWPAN. Nota-se no primeiro gráfico de barras que para uma página pequena a redução de *overhead* entre comprimir os cabeçalhos IP e não comprimir é grande – no caso de não haver compressão grande parte dos dados é dedicada apenas ao cabeçalho IP, mais especificamente 39.5% do total de dados enviados, contra 25% no caso da compressão. Lembrando que não é utilizada compressão de cabeçalho TCP, uma vez que não foi especificada de forma oficial no momento de realização deste trabalho.

[1] www.wireshark.org

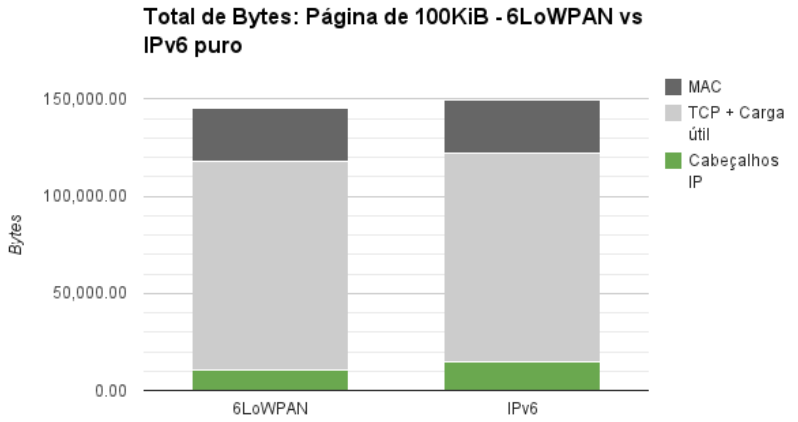


Figura 31 – Quantidade absoluta de bytes enviadas pelo servidor. Página de 100 KiB

Fonte: elaborada pelo autor

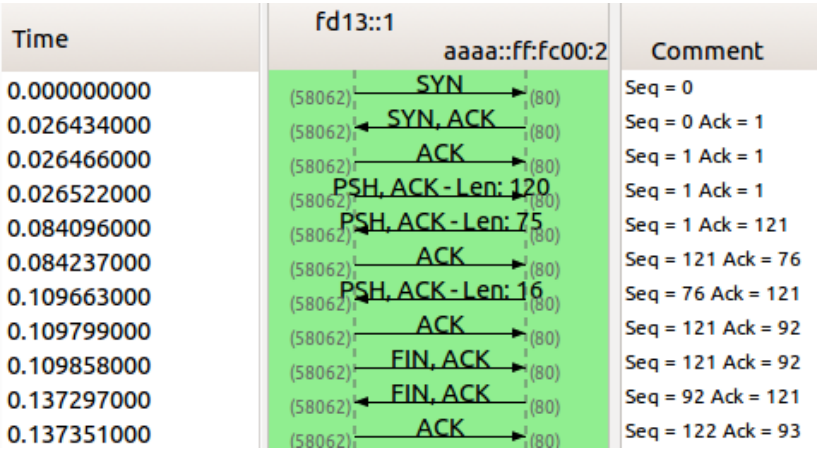


Figura 32 – Análise da troca de dados na requisição e resposta de uma página de 16 bytes

Fonte: elaborada pelo autor

Com testes utilizando páginas maiores a diferença cai em termos relativos ao total de bytes trocados. Quando comprimindo os dados, cerca de 7% do total de dados enviados corresponde a cabeçalho IP, contra 9.5% no caso de não haver compressão. Em termos absolutos, a quantidade de bytes trocados no caso da página de 16 bytes é de 1221 bytes com compressão contra 1507 bytes sem (286 bytes a menos quando usando compressão). No caso da página de 100 KiB tem-se 145316 bytes trocados quando usando compressão contra 149579 bytes sem compressão, ou seja, cerca de 4 KiB a menos quando utilizando compressão.

Em termos relativos tem-se um ganho maior com páginas menores – o que pode se tornar um ganho alto em termos absolutos caso as requisições para tal página sejam frequentes. Apesar do menor impacto com páginas maiores, a economia de transmissão de 4 KiB no caso da carga útil de 102400 bytes é significativa e justifica o uso do 6LoWPAN. Páginas de 100 KiB são problemáticas em dispositivos com poucos recursos – para o teste ser realizado tal página foi dividida em blocos de 1 KiB. Em uma situação real tal página poderia ser, por exemplo, gerada com dados sendo coletados em tempo real (uma vez que o maior tamanho de *buffer* em RAM possível para dados de aplicação não passa de 2 KiB no dispositivo utilizado).

A variação no tempo de carregamento entre os casos com e sem compressão é perceptível. Apesar de à primeira vista parecer pequena, a diferença é significativa ao se considerar a magnitude dos tempos envolvidos na troca de dados. Em aplicações com requisições frequentes de dados via HTTP a diferença de tempo constatada pode trazer benefícios em termos de consumo de energia, conforme promovido pelos documentos de especificação do 6LoWPAN.

6.3.5 Experimento 9: Rede de Sensores

6.3.5.1 Objetivo

O objetivo deste experimento é o de avaliar a rede desenvolvida, os dispositivos e o *gateway*, em um caso real de uso, com sensores enviando dados coletados a um computador na rede Ethernet. Deve-se verificar com e sem compressão IPHC/NHC quantas amostragens se consegue fazer de cada sensor em um mesmo intervalo de tempo, além do total de amostragens em todos os sensores, a fim de comprovar em uma estrutura comum de rede de sensores a efetividade do uso do 6LoWPAN.

A topologia da rede adotada é exibida na Figura 25. Um programa servidor executado no PC, implementado em Python, fazia requisição dos

dados dos sensores nos nós.

A quantidade de nós na rede sem fio foi variada entre 1 e 6 nós (além do nó do *gateway*). Cada nó correspondia a um sensor. Neste teste foi utilizado como valor de amostragem dos sensores a força de recepção do sinal de rádio, o RSSI. O PC submetia uma requisição de valor de RSSI para cada um dos sensores e o sensor imediatamente respondia. Ambos requisição e resposta possuíam mesmo tamanho de pacote e eram enviados via UDP. A carga útil de dados da aplicação era de 6 bytes por pacote. O programa no PC pedia o valor da amostragem do próximo sensor no ciclo imediatamente após receber a resposta do anterior. Os dados RSSI eram plotados na aplicação do PC assim que eram recebidos.

O programa no PC percorria circularmente todos os sensores da rede durante 10 segundos, armazenando estatísticas de dados trocados. Ao final dos 10 segundos as estatísticas de quantidade de pacotes enviados para cada sensor eram compiladas para verificação da quantidade de dados trocada. O teste foi realizado variando a quantidade de sensores. O intervalo de endereços de IP utilizado pelos nós foi do endereço aaaa::ff:fc00:2 ao aaaa::ff:fc00:7. Os mesmo testes, com os nós posicionados da mesma forma e com os mesmos endereços, foram realizados utilizando compressão IPHC/NHC e IPv6 sem compressão.

Foram realizados 10 testes para cada quantidade de nós variando entre 1 e 6. A média de pacotes transferidos com cada número de nós é registrada. São desconsiderados e refeitos testes em que há perda de pacotes.

O gráfico da Figura 33 exibe a quantidade total de amostras realizadas de todos os sensores juntos, considerando quantidade variável de sensores e comparando as abordagens com e sem compressão. Na Figura 34 a mesma comparação é realizada considerando a quantidade de amostras por sensor, individualmente.

Como conclusão observa-se que o uso do 6LoWPAN permite maior quantidade de amostragens por sensor e maior quantidade total de amostras na soma de todos os sensores por consequência. Quanto menor o pacote, maior a vantagem do uso do 6LoWPAN pois quanto mais cabeçalhos IP são enviados maior o impacto da compressão no total de dados trocados. A transmissão de menor quantidade de dados implica em menos energia sendo utilizada em *setup* com baixa quantidade de amostragens. Nestes casos, considerando o uso de técnicas de RDC, o rádio permanece menos tempo ligado para enviar a mesma quantidade de dados úteis.

A rede funcionou bem. Executou-se o teste algumas dezenas de vezes sem problemas. Isto mostra que os nós desenvolvidos em conjunto com o *gateway* podem ser utilizados nos cenários previstos na motivação deste trabalho, na presença de sensores e atuadores remotamente monitorados e

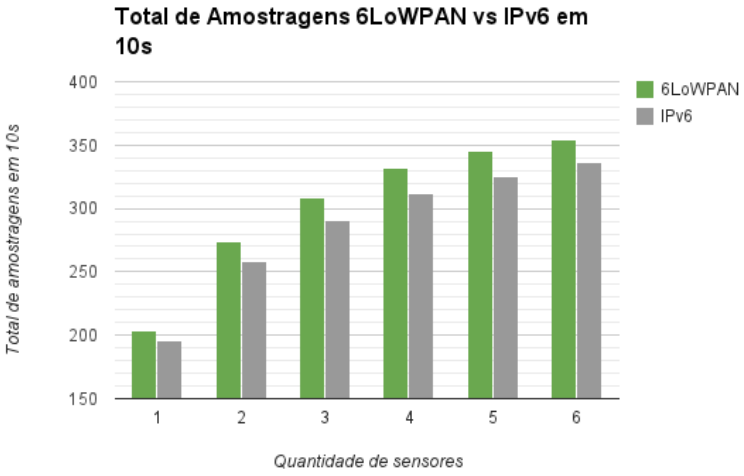


Figura 33 – Quantidade total de amostras em 10 segundos, considerando todos os sensores combinados em cada teste
Fonte: elaborada pelo autor

controlados. A percepção da possibilidade de maior troca de carga útil em um mesmo intervalo de tempo ficou clara ao se utilizar o 6LoWPAN, confirmando em uma situação prática a efetividade da técnica.

Notou-se nos testes variação dos resultados de acordo com ordem com que os sensores eram ligados – identificou-se relação com a forma como a árvore de roteamento de dados é formada no RPL. Após executar o comando de reparação da árvore no nó do *gateway* a árvore RPL é reconstruída e os resultados passam a apresentar repetibilidade entre um teste e outro. Para resultados mais consistentes foi também mantido o *setup* e posição dos nós durante todos os testes.

6.4 CONSIDERAÇÕES FINAIS

Os experimentos realizados, apesar de básicos, permitiram a comprovação de que os nós possuem desempenho adequado para aplicações em WPANs, a análise das características do padrão 6LoWPAN e o vislumbre de potenciais usos para os equipamentos desenvolvidos. Os resultados foram satisfatórios em termos de alcance e velocidade de comunicação. Pôde-

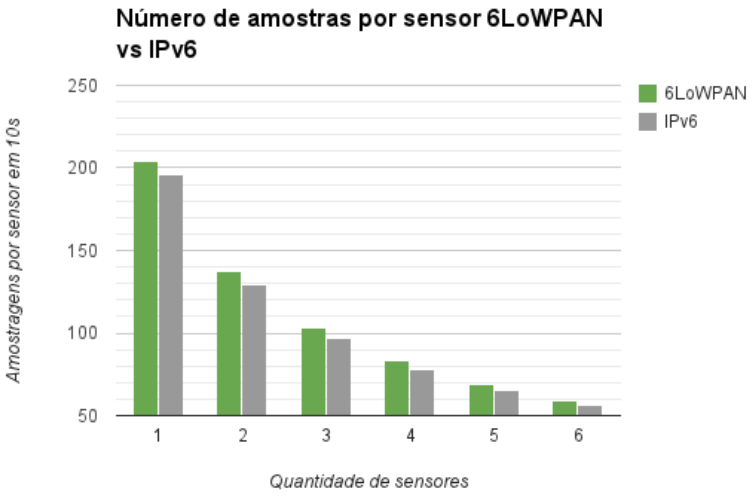


Figura 34 – Quantidade de amostras por sensor em 10 segundos
Fonte: elaborada pelo autor

se constatar a viabilidade da solução desenvolvida para usos em aplicações práticas.

7 CONCLUSÃO E PERSPECTIVAS

O objetivo principal do trabalho, de integrar uma rede WPAN de dispositivos de baixo custo a uma rede TCP/IP convencional Ethernet, foi atingido com êxito. A comunicação a partir da rede Ethernet foi realizada com sucesso com os nós da rede sem fio. Tanto a rede sem fio como o *gateway* funcionaram perfeitamente. A velocidade de comunicação entre os nós e o desempenho obtido com o hardware desenvolvido permitem que sejam utilizados em aplicações sem fio quase que de forma direta. O *gateway* desenvolvido, apesar da limitação da interface serial entre seus componentes e do seu formato de protótipo, já é passível de utilização em aplicações práticas.

A natureza modular dos dispositivos construídos permite que qualquer equipamento possa ser conectado à Internet. O uso do Sistema Operacional Contiki permitiu, de forma robusta, que diversas funcionalidades fossem incorporadas ao dispositivo de uma vez. Muitos recursos do Sistema Operacional ainda podem ser explorados. O sistema também possui muitas funcionalidades ainda não implementadas em se tratando de IEEE 802.15.4. A parte da especificação que define redes com controle de acesso baseado em *beacons*, por exemplo, não foi implementada. Os *slots* de tempo garantido, os quais aproximam as redes sem fio do uso em cenários industriais, também não estão presentes. Apesar de apresentar soluções como o ContikiMAC, tais funcionalidades permitiriam maior facilidade na integração com nós compatíveis com IEEE 802.15.4 que utilizam outros Sistemas Operacionais.

No *gateway*, o uso do Linux coloca o dispositivo em um patamar elevado de sistema embarcado, tendo comportamento semelhante a um computador. No escopo do trabalho o *gateway* teve apenas funcionalidades simples de roteamento implementadas, porém o uso de outros programas no Linux e a configuração de outros serviços podem permitir que as funcionalidades da rede sem fio sejam ampliadas. O *gateway* pode ser configurado, por exemplo, para gerenciar as amostragens dos sensores e fornecer uma interface HTTP para consulta externa.

As vantagens do uso do 6LoWPAN, e algumas situações em que seu uso não traz ganhos significativos, puderam ser comprovadas na prática durante o trabalho. A forma transparente como se integra a uma rede IPv6 o torna bastante adequado para LLNs, se provando uma alternativa forte para redes baseadas em IEEE 802.15.4 mais difundidas como o Zigbee® e WirelessHART® e que possui grandes chances de substituí-las, constituindo um padrão mais amplamente adotado em LLNs. O fato do IP ser dominante em termos de utilização no mundo colabora com esta perspectiva.

Uma outra contribuição deste trabalho foi a análise de desempenho dos

mecanismos de compactação de cabeçalhos e fragmentação do 6LoWPAN. Por ser um padrão recente, e por ser proposto para a integração de sistemas embarcados à Internet em aplicações do conceito de Internet das Coisas, é importante que seu desempenho seja suficiente para uma gama de aplicações que interagem diretamente com o ambiente, e por isso possuem restrições temporais.

Como principal perspectiva para trabalhos futuros tem-se um estudo mais aprofundado do Sistema Operacional e do hardware para permitir que o consumo de energia seja controlado e calculado, visando permitir o uso do dispositivo em cenários de pouca disponibilidade de energia, em que há uso de baterias e de técnicas de *energy harvesting*. Testes mais abrangentes com as diferentes camadas possíveis do Contiki devem ser realizados para chegar a uma configuração do sistema com melhor compromisso entre latência e consumo energético. O aumento da velocidade de conexão entre dispositivo com conectividade Ethernet e nó desenvolvido também deve ser objeto de estudo, valendo considerar inclusive a troca da interface serial por uma interface SPI, que permite troca de dados confiável a velocidades superiores.

O protótipo do *gateway* permitiu a validação do sistema, porém o uso de fios entre a Beaglebone e o módulo e o excesso de tamanho e periféricos desnecessários torna inviável seu uso sistemático ou mesmo comercial, até por se tratar de um protótipo. O desenvolvimento de um *gateway* integrado, compacto, e que concentre apenas as funções de *gateway* em si, o aproximando de um sistema embarcado dedicado, é desejado no futuro.

Neste trabalho foram dados os primeiros, e, possivelmente, mais importantes passos, no objetivo maior de desenvolver um *gateway* portátil compatível com o padrão 6LoWPAN, testado e validado em diferentes cenários de uso e cuja configuração seja simples a ponto do dispositivo poder ser configurado com a mesma facilidade que roteadores de redes sem fio 802.11 são hoje. Tal dispositivo, além de poder ser utilizado comercialmente após adequações para tal, pode facilitar a realização de trabalhos e pesquisas nas mais diversas áreas do conhecimento que exijam dispositivos pequenos com conectividade sem fio simples, barata, aberta e padrão, abrindo uma gama de possibilidades de implementação de redes.

O trabalho originou o artigo intitulado *Integration of Wireless Sensor Networks to the Internet of Things using a 6LoWPAN Gateway* (SCHRICKTE et al., 2013), aceito no III SBESC - Simpósio Brasileiro de Engenharia de Sistemas Computacionais.

REFERÊNCIAS

AFANASYEV, M. et al. Heterogeneous traffic performance comparison for 6lowpan enabled low-power transceivers. In: **Proceedings of the 6th Workshop on Hot Topics in Embedded Networked Sensors**. New York, NY, USA: ACM, 2010.

ASHTON, K. That 'internet of things' thing. **RFID Journal**, 2009.

ATMEL. **ATmega128RFA1**. Fevereiro 2013. Disponível em: <http://www.atmel.com/devices/atmega128rfa1.aspx>.

ATZORI, L.; IERA, A.; MORABITO, G. The internet of tihings: A survey. **Computer Networks**, 2010.

BUETTNER, M. et al. X-mac: a short preamble mac protocol for duty-cycled wireless sensor networks. In: **Proceedings of the 4th international conference on Embedded networked sensor systems**. New York, NY, USA: ACM, 2006. (SenSys 06), p. 307–320.

CAMPOS, B. da S. et al. Design and construction of wireless sensor network gateway with ipv4/ipv6 support. In: **IEEE International Conference on Communications (ICC)**. Kyoto, Japan: IEEE, 2011. p. 1–5.

CETIC. **6LBR**. 2013. Web site. Disponível em: <https://github.com/cetic/6lbr/wiki>.

COMPUTER SOCIETY. **IEEE Standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements - Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)**. 2006.

CRAWFORD, M. **Transmission of IPv6 Packets over Ethernet Networks**. Dezembro 1998.

DEERING, S.; HINDEN, R. **Internet Protocol, Version 6 (IPv6) Specification**. Dezembro 1998.

DUNKELS, A. Rime - a lightweight layered communication stack for sensor networks. In: **Proceedings of the European Conference on Wireless Sensor Networks (EWSN), Poster/Demo session**. Delft, The Netherlands:

[s.n.], 2007. Disponível em:
<http://dunkels.com/adam/dunkels07rime.pdf>.

DUNKELS, A. **The ContikiMAC Radio Duty Cycling Protocol**. [S.l.], 2011.

DUNKELS, A. et al. Poster abstract: Making sensor networks ipv6 ready. **SenSys 08**, 2008.

DURST, L. **Standards are Making the Internet of Things Come Alive**. 2013. Disponível em:
<http://standardsinsight.com/ieee_company_detail/standards_iot>.

FAROOQ, M. O.; KUNZ, T. Contiki-based ieee 802.15.4 nodeś throughput and wireless channel utilization analysis. **Wireless Days (WD), IFIP**, p. 1–3, 2012.

GUSMEROLI, S. et al. **Vision and Challenges for Realising the Internet of Things**. Luxembourg: European Commission, 2010.

HART COMMUNICATION FOUNDATION. 2013. Disponível em:
<www.hartcomm.org>.

HUI, J.; CULLER, D.; CHAKRABARTI, S. **6LoWPAN: Incorporating IEEE 802.15.4 into the IP architecture**. [S.l.], 2009.

HUI, J.; THUBERT, P. **Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks**. 2011.

JIANG, X. et al. Design and implementation of a high-fidelity ac metering network. In: **Proceedings of the 2009 International Conference on Information Processing in Sensor Networks**. [S.l.]: IEEE Computer Society, 2009. (IPSN '09), p. 253–264.

KO, J. et al. Beyond interoperability - pushing the performance of sensor network ip stacks. In: **Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems**. New York, NY, USA: ACM, 2011. (SenSys 11), p. 1–11.

KO, J. et al. Contikirpl and tinyrpl: Happy together. In: **In Proceedings of the workshop on Extending the Internet to Low power and Lossy Networks (IP+SN 2011)**. Chicago, Illinois, USA: ACM, 2011.

KOUMPIS, K. et al. **Wireless Industrial Control and Monitoring beyond Cable Replacement**. Junho 2005.

KUSHALNAGAR, N.; MONTENEGRO, G.; SCHUMACHER, C. Internet Standard, **IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement and Goals**. Agosto 2007.

LUDOVICI, A. et al. The internet of the future. In: OLIVER, M.; SALLENT, S. (Ed.). **Proceedings of the 15th Open European Summer School and IFIP TC6.6 Workshop on The Internet of the Future**. Barcelona, Espanha: Springer Berlin Heidelberg, 2009. v. 5733, p. 168–177.

MAYER, K.; FRITSCHE, W. Ip-enabled wireless sensor networks and their integration into the internet. In: **Proceedings of the first international conference on Integrated internet ad hoc and sensor networks**. New York, NY, USA: ACM, 2006. (InterSense '06). ISBN 1-59593-427-8.

MONTENEGRO, G. et al. **Transmission of IPv6 Packets over IEEE 802.15.4 Networks**. Setembro 2007.

MULLIGAN, G. The 6lowpan architecture. In: **Proceedings of the 4th workshop on Embedded networked sensors**. New York, NY, USA: ACM, 2007. (EmNets '07), p. 78–82.

NARTEN, T. et al. Draft Standard, **Neighbor Discovery for IP version 6 (IPv6)**. Setembro 2007.

PRETZ, K. **Exploring the Impact of the Internet of Things**. Piscataway, NJ, USA, 2013.

ROMKEY, J. **A Nonstandard for Transmission of IP Datagrams Over Serial Lines: SLIP**. Junho 1988.

SCHRICKTE, L. F. et al. Integration of wireless sensor networks to the internet of things using a 6lowpan gateway. In: **SBESC 2013 - Critical Systems Track ()**. Niteroi, RJ, Brasil: [s.n.], 2013.

SHELBY, Z.; BORMANN, C. **6LoWPAN: The Wireless Embedded Internet**. NJ, USA: Wiley, 2009.

SHELBY, Z.; CHAKRABARTI, S.; NORDMARK, E. Draft Standard, **Internet Draft Standard - Neighbor Discovery Optimization for Low Power and Lossy Networks (6LoWPAN)**. Agosto 2012.

SILVA, R.; SILVA, J. S.; BOAVIDA, F. Evaluating 6lowpan implementations in wsns. In: **Conferencia sobre Redes de Computadores**. [S.l.: s.n.], 2009.

SINNIAH, G. R. et al. Ipv6 wireless sensor network gateway design and end-to-end performance analysis. In: **SENSORCOMM 2012: The Sixth International Conference on Sensor Technologies and Applications**. [S.l.: s.n.], 2012.

TEXAS INSTRUMENTS AND DIGIKEY. **BeagleBoard Project Website**. 2008. Disponível em: <http://beagleboard.org>.

THE CONTIKI PROJECT AND ITS CONTRIBUTORS. **Contiki: The Open Source Operating System for the Internet**. 2003. Disponível em: <http://www.contiki-os.org/>.

THOMSON, S.; NARTEN, T.; JINMEI, T. Draft Standard, **IPv6 Stateless Address Autoconfiguration**. Setembro 2007. Disponível em: <RFC4862>.

VASSEUR, J. et al. **RPL: The IP routing protocol designed for low power and lossy networks**. [S.l.], 2011.

WILLIG, A.; MATHEUS, K.; WOLISZ, A. Wireless technology in industrial networks. In: **Proceedings of the IEEE**. [S.l.]: IEEE, 2005. v. 93, p. 1130–1151.

WINTER, T. et al. **RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks**. Marco 2012.

ZIGBEE ALLIANCE. **Zigbee**. 2013. Disponível em: <www.zigbee.org>.