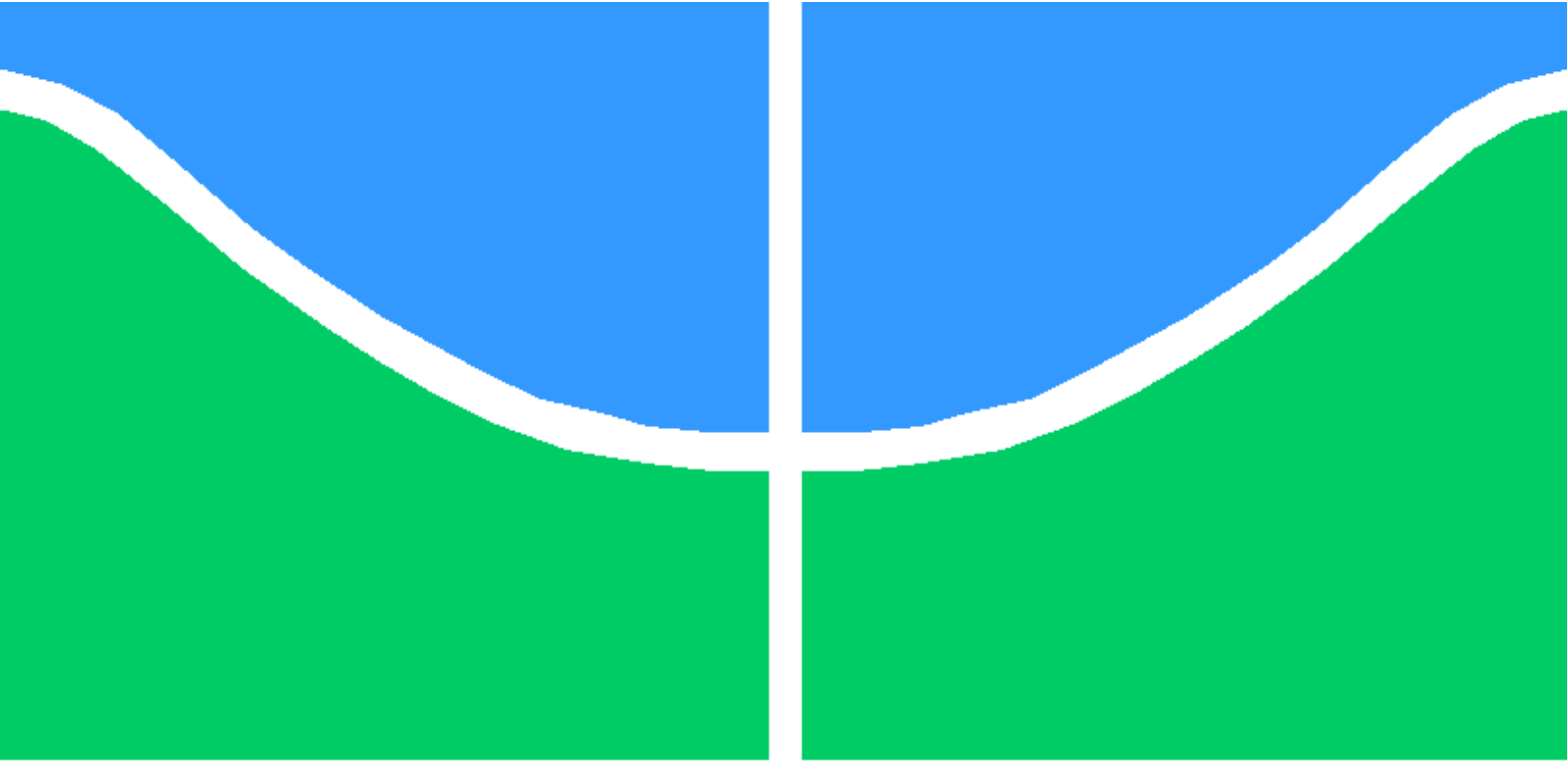

TRABALHO DE GRADUAÇÃO

Proposta e Implementação de extensão de autenticação de Manet para protocolo OLSR.

Humberto Freitas Araújo
Vinícius Campos de Paula

Brasília, janeiro de 2010

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASILIA
Faculdade de Tecnologia

TRABALHO DE GRADUAÇÃO

Proposta e Implementação de extensão de autenticação de Manet para protocolo OLSR.

Humberto Freitas Araújo

Vinícius Campos de Paula

Relatório submetido ao Departamento de Engenharia Elétrica
como requisito parcial para obtenção do grau de
Engenheiro de Redes de Comunicação

Banca Examinadora

Prof. Dr. Ricardo Staciarini Puttini, ENE/UnB (Orientador) _____

Prof. Dr. Anderson Clayton Alves Nascimento, ENE/UnB (Membro Interno) _____

MSc. Yamar Aires da Silva (Membro Externo) _____

FICHA CATALOGRÁFICA

ARAÚJO, H. F.. DE PAULA, V. C.
Proposta e Implementação de extensão de autenticação de Manet para protocolo OLSR.
[Distrito Federal] 2010.
v, 35p. (ENE/FT/UnB, Engenheiro de Redes de Comunicação, 2010)
Monografia de Graduação - Universidade de Brasília. Faculdade de Tecnologia.
Departamento de Engenharia Elétrica.
1. MANET 2. OLSRD
3. Autenticação 4. Roteamento
I. ENE/FT/UnB II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA

ARAÚJO, H. F. (2010) e DE PAULA, V. C. (2010). Proposta e Implementação de extensão de autenticação de Manet para protocolo OLSR. Monografia de Graduação, Publicação ENE 01/2010, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 35p.

CESSÃO DE DIREITOS

NOMES DOS AUTORES: Humberto Freitas Araújo e Vinícius Campos de Paula.

TÍTULO: Proposta e Implementação de extensão de autenticação de Manet para protocolo OLSR.

GRAU / ANO: Engenheiro de Redes de Comunicação / 2010.

_____

Humberto Freitas Araújo

_____

Vinícius Campos de Paula

Dedicatórias

Dedico este trabalho a minha avó, aos meus pais, meus irmãos, meus amigos e minha adorável sobrinha Júlia.

Humberto Freitas Araújo

Dedico este trabalho a minha família, meus amigos e minha namorada..

Vinícius Campos de Paula

Agradecimentos

Agradeço meu professor e orientador Ricardo Puttini pela orientação, a todos os professores do departamento que me ajudaram na busca pelo conhecimento, aos meus amigos que tornaram mais agradável a passagem pela universidade e minha família por todo apoio necessário.

Humberto Freitas Araújo

Agradeço meu professor e orientador Ricardo Puttini pela orientação neste trabalho, a todos os professores que tive na Universidade de Brasília que me ajudaram durante todo este período, a minha família que sempre me apoiou nesta caminhada, aos meus amigos e principalmente a turma de Engenharia de Redes 04/02 a qual passamos grandes momentos.

Vinícius Campos de Paula

RESUMO

Neste trabalho é apresentado uma proposta e implementação de uma extensão de autenticação para MANET. Propõem-se um serviço de autenticação em nível de roteamento para as mensagens do protocolo de roteamento OLSR - Optimized Link State Routing que está documentado na RFC 3626 em sua primeira versão. Adaptou-se a formatação de pacotes e mensagens para MANET que foi projetado para trasportar múltiplas mensagens e está definida na RFC 5444. Um dos serviços de segurança com caráter preventivo é a MAE - MANET Authtentication Extension, que prove uma extensão de autenticação possibilitando assegurar que as mensagens de MANETs advindas de nodos não confiáveis recebam tratamento de acordo com a política de segurança, podendo ser simplesmente descartadas ou tratadas de forma diferente. Este trabalho tem como objetivo implementar a MAE para OLSR v1.

ABSTRACT

This work presents a proposition and implementation of Manet Authentication Extension (MAE). It is proposed an authentication service in routing level including routing messages of OLSR which is presented by the first version of RFC 3626. The packet and messages formats were adapted to fit multiples messages according to RFC 5444. One of the security service with preventive nature is MAE, which provides an authentication extension ensuring that MANET messages coming from untrustful nodes receive different treatment according to the security politic. The message may be rejected or handled in a different way. The work's objective is to implement MAE for OLSR v1.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

Lista de Acrônimos

ANSN	*Advertised Neighbor Sequence Number*
BSD	*Berkeley Software Distribution*
DS	*Digital Signature*
HC	*Hash Chain*
HNA	*Host and Network Association*
IANA	*Internet Assigned Numbers Authority*
IP	*Internet Protocol*
MAC	*Message Authentication Code*
MAE	*Manet Authentication Extension*
MANET	*Mobile Ad Hoc Networks*
MID	*Multiple Interface Declaration*
MPR	*Multipoint Relay*
OLSR	*Optimized Link State Routing Protocol*
OLSRD	*Optimized Link State Routing Protocol Daemon*
RFC	*Request for Comments*
SSL	*Secure Sockets Layer*
SPF	*Shortest Path First*
TC	*Topology Control*
TLV	*Type Length Value*
TTL	*Time To Live*
UDP	*User Datagram Protocol*
VTIME	*Validation Time*

1 INTRODUÇÃO

(*Mobile Ad Hoc Network* - MANET) é uma coleção de nodos móveis sem fio que cooperativamente formam uma rede sem infra-estrutura fixa. Em redes Ad Hoc é possível que dispositivos criem uma rede por demanda sem necessidade de uma coordenação ou configuração. Para que isso seja possível os nodos da rede estão envolvidos no roteamento e no encaminhamento de pacotes entre os vizinhos. Dessa maneira, quando um nodo precisa se comunicar com outro que não esteja no alcance do seu enlace, ele encaminha sua informação através de um nodo vizinho e assim segue até a informação chegar ao destinatário[1].

Algumas aplicações da tecnologia MANET podem incluir aplicações industriais e comerciais envolvendo troca de informações móveis. Existe também um futuro em aplicações militares que exigem robustez e serviços de informação com redes de comunicação sem fio móvel. Pode-se combinar à entrega de informação via satélite, provendo um método de grande flexibilidade para estabelecer comunicações com operações de salvamento, incêndio, segurança ou qualquer outro cenário onde é necessária a comunicação dinâmica[2].

As comunicações sem fio estão sujeitas a problemas referentes à largura de banda limitada dos enlaces, efeitos de propagação multipercurso, efeitos de desvanecimento seletivo em freqüência (Fading), sombreamento (Shadowing) e interferências. Como os nodos que constituem as MANETs são móveis, a topologia de rede é extremamente dinâmica e aleatória, desse modo, os serviços em uma MANET devem seguir um modelo descentralizado[3]. Algumas vulnerabilidades acompanham as comunicações sem fio:

- Escuta do canal de comunicação por dispositivos de escuta os quais estejam no raio de alcance dos transmissores sem fio.
- Comunicação direta entre dois nodos próximos
- Mobilidade dos nodos, permitindo que um nodo saia do alcance de um conjunto de nodos e entre no alcance de outro conjunto.
- Não-colaboração dos nodos (para evitar exaustão de recursos próprios ou para provocar disfunções no encaminhamento de pacotes na rede).
- Utilizar os recursos da rede com objetivo de exaurir a alimentação de outros nodos.

Muitas das vulnerabilidades de segurança que ocorrem nas arquiteturas de redes tradicionais podem ocorrer

em MANETs e ainda podem ser exploradas de formas diferentes. Outras vulnerabilidades ocorrem dadas a natureza das redes MANET. Um dos problemas é a integridade da rede Ad Hoc, de acordo com suas especificações, permite qualquer nodo participar da rede, assumindo que o mesmo é bem comportado. Se essa assunção falhar a rede pode estar sujeita a nodos maliciosos colocando em risco a integridade da rede. Os serviços básicos, assim como outros serviços de rede, são providos de forma descentralizada e com participação de todos os nodos da rede de forma colaborativa. Portanto, a falta de colaboração de alguns nodos ou por mau funcionamento ou para economia de energia/bateria, pode comprometer as funcionalidades da rede. Um nodo comprometido é aquele era tido como confiável pela rede e em algum momento começa apresentar um comportamento anormal. Esse comportamento inadequado advém de falhas intencionais ou não, além de influência por agentes externos. Algumas dos ataques que são exploradas no serviço de roteamento[3]:

– Modificação: Mensagens são modificadas entre a origem e o destino.

– Personificação: Um nodo pode se passar por outro para enviar mensagens na rede.

– Fabricação: Um nodo gera mensagens falsas.

– Não-Colaboração: Quando um nodo tem a obrigação de colaborar no roteamento e não o faz.

Os ataques em protocolos de roteamento já citados (modificação, fabricação e personificação) são encontrados nos protocolos de roteamento e ainda podem ser combinados aumentando as possibilidades de ataques. Esses ataques podem ser mitigados através de serviços de autenticação e confidencialidade. O primeiro permite que se verifique a autenticidade do transmissor da mensagem e o segundo garante a integridade da mesma.

Tendo em vista as características de uma rede Ad Hoc o serviço de roteamento está relacionado com a natureza multi-salto, portanto, o protocolo de roteamento deve levar em consideração as freqüentes mudanças na topologia de rede. O protocolo OLSR foi projetado para lidar com essa natureza das redes Ad Hoc e prova ser um protocolo robusto para configuração das redes MANETs atuais. Um software que implementa o protocolo OLSR[4] é o OLSRD[5] que pode ser encontrado na página `http://www.olsr.org`.

Visto que há uma preocupação cada vez maior em relação a segurança de MANETs, este trabalho tem como objetivo uma proposta e implementação de uma extensão de autenticação para MANET - MAE utilizando o OLSR conforme a formatação de pacotes especificada na RFC 5444[6]. A autenticação é

orientada a mensagens do protocolo de roteamento OLSR e opcionalmente mensagens de um serviço de certificação. A MAE deve conter todos os campos de informação do OLSR necessários para garantir a autenticidade e integridade dessas informações, protegendo contra os ataques de modificação, fabricação e personificação. Para isso foi desenvolvido uma extensão de autenticação para MANET em forma de plugin OLSRD.

Este trabalho foi organizado da seguinte maneira:

- Capítulo 2: Protocolo OLSR

  Funcionalidade do protocolo;

  Características.

- Capítulo 3: MAE para OLSR

  Segurança para OLSR;

  Sugestão de autenticação e integridade proposta pela RFC 5444.

- Capítulo 4: Implementação

  OLSRD;

  Implementação do plugin MAE

- Capítulo 5: Conclusão

  Resultados obtidos;

  Considerações finais.

2 PROTOCOLO OLSR

O protocolo OLSR foi desenvolvido especificamente para redes Ad Hoc e está documentada na RFC 3626 em caráter experimental. Ele é uma otimização do algoritmo baseado em estados de enlace e é pró-ativo, ou seja, informações da topologia da rede são trocadas entre os nodos regularmente agilizando a comunicação quando requisitada. Um elemento importante é o conceito de Multipoint Relays (MPRs)[4] que são os nodos eleitos e responsáveis por encaminhar o tráfego de controle que é difundido por toda rede. A RFC 3626 divide o OLSR em Funcionalidade Núcleo, que são as requisições necessárias para o protocolo operar, e um conjunto de Funções Auxiliares.

2.1 FUNCIONALIDADE NÚCLEO

É especificado o comportamento de um nodo equipado com interfaces OLSR participante de uma MANET executando OLSR como protocolo de roteamento, incluindo ainda uma especificação geral das mensagens OLSR e suas transmissões pela rede, assim como, a difusão da topologia de rede e cálculo de rota. A seguir os componentes que compõem o núcleo[4][5][7].

2.1.1 Endereçamento

OLSR utiliza o endereço IP como único identificador dos nodos na rede. Cada nodo deve escolher o seu endereço IP principal, pois o OLSR foi projetado para funcionar em nodos com múltiplas interfaces. Pode-se utilizar o OLSR tanto no IP versão 4 (IPv4)[8] quanto no IP versão 6 (IPv6)[9]. Em um contexto OLSR a diferença entre IPv4 e IPv6 é o tamanho do endereço IP transmitido nas mensagens de controle do protocolo.

2.1.2 Repositório de Informações

O protocolo OLSR advém do algoritmo de estados de enlace clássico e mantém os estados localmente uma variedade de bases de informação. Esses Repositórios de Informações são atualizados na medida em que as mensagens de controle são recebidas e a informação armazenada é usada para geras essas mensagens. Os diferentes Repositórios de Informações seguem abaixo:

– **Base de informações sobre Associações de Múltiplas Interfaces**

  Este repositório contém informações sobre os nodos que utilizam mais de uma interface para a comunicação. Os endereços de cada interface são armazenados aqui.

– **Base de Informações de Enlaces Locais**

  Registro de enlaces entre um nodo local e seus vizinhos.

– **Conjunto de Enlaces**

  Mantém os estados de enlace dos vizinhos. Esta base de dados opera em enlaces específicos de interface para interface, diferentemente das outras que utilizam os endereços principais dos nodos.

– **Conjunto de Vizinhos**

  Todos os vizinhos de um salto são armazenados aqui. Utiliza-se o Conjunto de Enlaces para atualizar dinamicamente a informação. Tanto os nodos simétricos, nodos os quais para cada existe um enlace simétrico para esse nodo em alguma interface, quanto nodos assimétricos são registrados aqui.

– **Conjunto de Vizinhos a dois saltos**

  São registrados os nodos que são vizinhos a dois saltos, que podem ser alcançados por vizinhos de um salto.

– **Conjunto de MPRs**

  Todos os MPRs selecionados pelo nodo são registrados nesse repositório. O conceito de MPR vai ser abordado posteriormente.

– **Conjunto de Seletores MPR**

  Registro de todos os vizinhos que selecionaram um nodo local como MPR.

– **Base de Informações Topológicas**

  Este repositório contém dados de todos os estados de enlace recebidos de todos os nodos no domínio de roteamento do OLSR.

– **Base de Informações de Duplicatas**

  Armazena informações sobre mensagens processadas e encaminhadas.

– **Base de Informações de Vizinhança**

  Registro de vizinhos, vizinhos de dois saltos, MPRs e seletores de MPRs.

2.1.3 Formato do pacote

A RFC 3626[4] define como é feita a comunicação no OLSR e formatação dos pacotes. Para facilitar a extensão do protocolo, o OLSR comunica utilizando um formato unificado de pacote para todo tipo de informação relacionada a ele. Os pacotes são transmitidos via datagramas UDP[10] pela porta 698 designada pela IANA (Internet Assigned Numbers Authority) e cada pacote encapsula uma ou mais mensagens. Todas as mensagens compartilham de um mesmo formato de cabeçalho, possibilitando que os nodos aceitem e retransmitam mensagens do tipo desconhecidas. As mensagens são enviadas em broadcast para toda rede, porém, mensagens de controle duplicadas são eliminadas localmente pelo uso da Base De Informação de Duplicatas e minimizadas em toda rede utilizando o conceito de MPRs. O modelo de qualquer pacote OLSR está na figura 2.1:

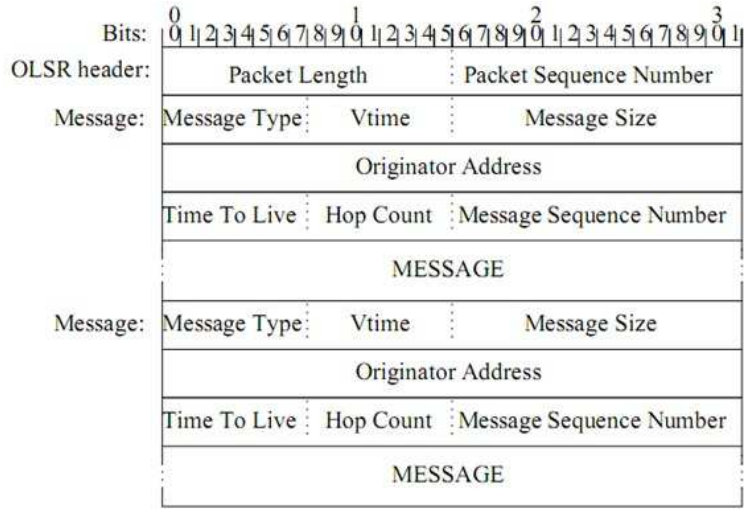

Figura 2.1: Modelo de Pacote OLSR.

A descrição dos campos do cabeçalho OLSR segue na tabela 2.1.

Campo	Descrição
Packet Length	O tamanho em bytes do pacote inteiro, incluindo o tamanho do cabeçalho.
Packet Sequence Number	Um número de seqüência incrementado por um cada vez que uma nova mensagem OLSR é transmitida pelo nodo. Cada interface de um nodo utiliza um número de seqüência independente.

Tabela 2.1: Cabeçalho OLSR.

Cada mensagem incluída no pacote OLSR tem o seu próprio cabeçalho, a descrição dos campos de uma mensagem OLSR segue na tabela 2.2.

Campo	Descrição
Message Type	Um número inteiro que identifica o tipo da mensagem. Tipos de mensagem de 0-127 são reservados para o OLSR enquanto 128-255 é privado e pode-se utilizar em extensões do protocolo.
Vtime	Indica o tempo após a recepção que um nodo vai considerar a informação contida no pacote como válida. Esse intervalo de tempo é calculado por sua mantissa-exponencial.
Message Size	Tamanho da mensagem em bytes, incluindo o cabeçalho da mensagem.
Originator Address	Endereço principal da origem da mensagem.
Time To Live	O número máximo de saltos que esta mensagem pode ser encaminhada. Valor deve ser decrementado por 1 quando um nodo recebe a mensagem.
Hop Count	Número de saltos esta mensagem realizou. Valor deve ser incrementado por 1.
Message Sequence Number	Cada mensagem gerada por um nodo é atribuído um número de seqüência a ela, este número deve ser incrementado de 1 a cada nova mensagem criada.

Tabela 2.2: Cabeçalho da mensagem OLSR.

2.1.4 Multipoint Relaying

Para difusão da topologia de rede, o OLSR utiliza a técnica de inundação (flooding) de pacotes, ou seja, todos os nodos retransmitem os pacotes recebidos. No entanto, o encaminhamento das retransmissões pode ocorrer de forma duplicada aumentando o uso da largura de banda do canal desnecessariamente. O conceito de multipoint relaying reduz este problema restringindo um conjunto de nodos responsáveis pela retransmissão para todos os outros nodos. Ocorre uma seleção de vizinhos como Multipoint Relays (MPRs). Cada nodo calcula o seu conjunto de MPRs sendo que devem ser nodos vizinhos, simétricos e que todos os vizinhos de dois saltos possam ser alcançados pelo MPR. Para cada nodo da rede que pode ser alcançado por um nodo local com dois saltos no mínimo, deve existir um MPR onde o nodo da rede tem um enlace simétrico com o MPR e o MPR é um vizinho simétrico do nodo local. Para facilitar o entendimento a figura 2.2 representa a técnica de flooding com MPRs em uma sem fio multisalto. Em vermelho o nodo local e amarelo os MPRs.

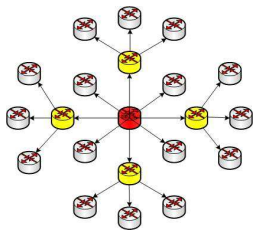

Figura 2.2: Técnica de Flooding com MPR.

2.1.5 Encaminhamento do tráfego OLSR

OLSR implementa um algoritmo padrão de encaminhamento de mensagens e utiliza o esquema de MPRs. Esse algoritmo se aplica a mensagens OLSR do tipo desconhecidas, para tipos de mensagens conhecidos o processamento é feito de acordo com a implementação local. O algoritmo é elencado da seguinte forma:

1. Se o enlace o qual a mensagem chegou não for simétrico, a mensagem é silenciosamente descartada. O Conjunto de Enlaces deve ser consultado.
2. Se o tempo de vida (TTL) do cabeçalho da mensagem é 0, a mensagem é silenciosamente descartada.
3. Se a mensagem já foi retransmitida é descartada. A Base de Informações de Duplicatas é consultada.
4. Se o último remetente da mensagem escolher este nodo como MPR, a mensagem é encaminhada. Se não, a mensagem é descartada. O Conjunto de Seletores MPR é consultado.
5. Se a mensagem deve ser encaminhada, o campo TTL deve ser decrescido de 1 e o campo Hop Count acrescido de 1 antes de enviar via broadcast para todas interfaces.

A RFC 3626 especifica três tipos de mensagens que estão incluídas na Funcionalidade Núcleo.

- **Mensagens HELLO:** Responsável pela detecção de enlaces, vizinhos e controle MPR.
- **Mensagens TC:** Responsável pela declaração da topologia de rede.
- **Mensagens MID:** Responsável pela declaração de múltiplas interfaces em um determinado nodo.

2.1.6 Múltiplas Interfaces

Quando um nodo possui apenas uma interface, o endereço principal é a da própria interface. No entanto, quando o nodo possui mais de uma interface, o endereço das interfaces é difundido pela mensagem MID (Multiple Interface Declaration). No processamento das mensagens MID ocorre atualização da Base de Informações sobre Associações de Múltiplas Interfaces. Quando um nodo recebe uma mensagem MID registram-se todos os endereços contidos na mensagem e o endereço principal encontra-se no campo Originator Address do cabeçalho. O formato da mensagem MID é a seguinte:

Nos campos OLSR Interface Address devem conter os endereços das outras interfaces. Os campos da mensagem OLSR devem ser preenchidos da seguinte maneira:

OLSR Interface Address
OLSR Interface Address

Figura 2.3: Modelo da mensagem MID.

– **MESSAGE TYPE:** MID MESSAGE

– **TTL:** 255

– **Vtime:** MID HOLD TIME

Resumindo, o cabeçalho ficaria da seguinte forma:

Packet Length		Packet Sequence Number
MID_MESSAGE	**MID_HOLD_TIME**	Message Size
MAIN ADDRESS		
255	Hop Count	Message Sequence Number
OLSR Interface Address		
OLSR Interface Address		

Figura 2.4: Modelo da Mensagem MID.

Os campos TTL e Hop Count são os únicos mutáveis na mensagem MID.

2.1.7 Mensagem HELLO

O mecanismo para popular a Base de Informações da Vizinhança e a Base de Informações de Enlaces Locais é a troca de mensagens HELLO entre os nodos. As mensagens HELLO são geradas e transmitidas para todos os vizinhos de um salto para detecção de estados de enlace, vizinhos, vizinhos de dois saltos e para seleção de MPR. Informações de todos os enlaces e vizinhos conhecidos são transmitidas na mensagem HELLO, incluindo quais MPRs um nodo elegeu.

Para a detecção de enlaces, vizinhos e seleção de MPR o formato da mensagem HELLO proposto pela RFC 3626 pode ser visto na figura 2.5

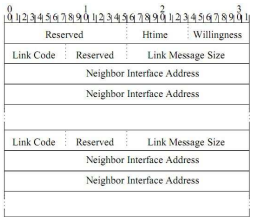

Figura 2.5: Formato da Mensagem HELLO.

Esta mensagem é enviada no pacote OLSR descrito anteriormente na figura 2.1, como parte da informação, porém, o campo Message Type deve ser HELLO MESSAGE e o campo TTL deve ser 1, pois a

mensagem HELLO não é encaminhada. Os campos da mensagem HELLO são descritos na tabela 2.3

Campo	Descrição
Reservado	Deve ser ajustado para 0000000000000 para ficar de acordo com a especificação.
Htime	É o intervalo de emissão de mensagens HELLO por interface do nodo. Este valor é calculado por sua Mantissa.
Willingness	Especifica a vontade do nodo na participação do encaminhamento do tráfego para outros nodos.
Link Code	Informação sobre o enlace entre a interface do remetente e a lista de vizinhos especificada no campo Neighbor Interface Address.
Link Message Size	Tamanho da mensagem de enlace medido do início do campo Link Code até o próximo campo Link Code, caso não exista mais nenhum link code, a medida vai até o final da mensagem.
Neighbor Interface Address	Endereço IP da interface de um nodo

Tabela 2.3: Campos de Mensagem HELLO.

A mensagem HELLO tem três objetivos já citados anteriormente:

– **Detecção de Enlace**

– **Detecção da Vizinhança**

– **Seleção de MPR**

Essas informações são providas pela troca periódica de mensagens HELLO entre a vizinhança. As mensagens HELLO são geradas a partir de consultas nos repositórios Conjunto de Enlaces, Conjunto de Vizinhos e Conjunto de MPRs, que estão na Base de Informações de Enlaces Locais.

Um nodo deve prover a detecção do enlace em cada interface entre cada interface vizinha. Posteriormente, um nodo deve anunciar sua presença para toda sua vizinhança simétrica de um salto em cada interface para detecção da vizinhança. Enfim, para uma dada interface, uma mensagem HELLO conterá uma lista de enlaces simétricos nessa interface, assim como uma lista de toda vizinhança.

2.1.7.1 Detecção de Enlace

A Detecção de Enlace se concentra exclusivamente no endereço da interface OLSR e na habilidade de trocar pacotes entre essas interfaces. Este processo faz com que o Conjunto de Enlaces fique povoado e atualizado pela troca de mensagens HELLO entre os nodos. Cada nodo detecta o enlace entre ele e seus vizinhos e verifica se os enlaces são bidirecionais. Caso afirmativo o enlace é dado como simétrico, caso negativo o enlace é dado como assimétrico, ou seja, a comunicação do nodo vizinho é possível, mas não se pode garantir que o nodo vizinho receberá mensagens. Na figura 2.6 ilustra como é feito a detecção de um enlace simétrico entre nodos vizinhos utilizando mensagens HELLO[5].

1. Nodo X envia uma mensagem HELLO vazia;

2. Nodo Y recebe a mensagem e registra o nodo X como vizinho assimétrico, isso ocorre, pois o nodo Y não encontrou seu endereço na mensagem HELLO;

3. Nodo Y envia mensagem HELLO declarando X como vizinho assimétrico e quando X receber encontrará seu endereço na mensagem tornando o nodo Y simétrico;

4. Nodo X inclui o endereço de Y na mensagem HELLO e envia, então Y registra X como vizinho simétrico.

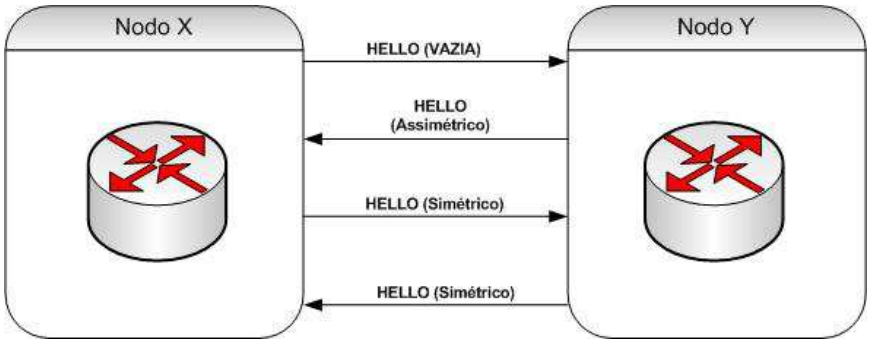

Figura 2.6: Detecção de Enlace Simétrico.

2.1.7.2 Detecção da Vizinhança

A Detecção da Vizinhança se concentra nos nodos e nos endereços principais de cada nodo. O mecanismo utilizado para detecção dos nodos vizinhos é a troca de mensagem HELLO entre os nodos, povoando e atualizando a Base de Informações de Vizinhança. Existe uma relação direta entre o repositório Conjunto de Enlaces e o Conjunto de Vizinhos, pois um nodo é vizinho de outro nodo se e somente se existir no mínimo um enlace entre os dois nodos. Toda vez que um enlace é criado, a lista de vizinhos é consultada para encontrar uma entrada de um vizinho correspondente. Essa entrada de um vizinho deve ser registrada com o endereço principal do nodo, se nenhuma entrada é localizada, cria-se então uma nova entrada do vizinho. Um nodo pode ter várias entradas de enlace uma diferente da outra para um mesmo vizinho, porém, existe apenas uma entrada de vizinho para cada vizinho, clarificando a relação entre os repositórios.

A detecção de vizinhos de dois saltos é feita pela manutenção do repositório Conjunto de Vizinhos a Dois Saltos, onde se encontra todos os vizinhos que podem ser alcançados por vizinhos simétricos. Quando se recebe uma mensagem HELLO de um vizinho simétrico para cada endereço nos campos Neighbor Interface Address é adicionada uma entrada no repositório.

2.1.7.3 Seleção de MPR

Cada nodo mantém um conjunto de vizinhos que são selecionados como MPRs. Seus endereços ficam listados no repositório Conjunto de MPRs. Um nodo também registra um Conjunto de Seletores MPRs que contém informações sobre um vizinho que o escolheu como MPR. Essas informações trazem o endereço do nodo vizinho que o escolheu como MPR e o tempo de expiração.

2.1.8 Descobrimento da Topologia

A detecção de enlaces e a detecção da vizinhança oferecem para cada nodo uma lista de vizinhos os quais eles podem se comunicar diretamente e encaminhar essas informações com o mecanismo de flooding pelos MPRs. Assim o estado de enlace do nodo com seus vizinhos é disseminado por toda rede e utilizado para o cálculo de rotas. Isso é feito utilizando a mensagem TC (Topology Control), que pode ser vista na figura 2.7.

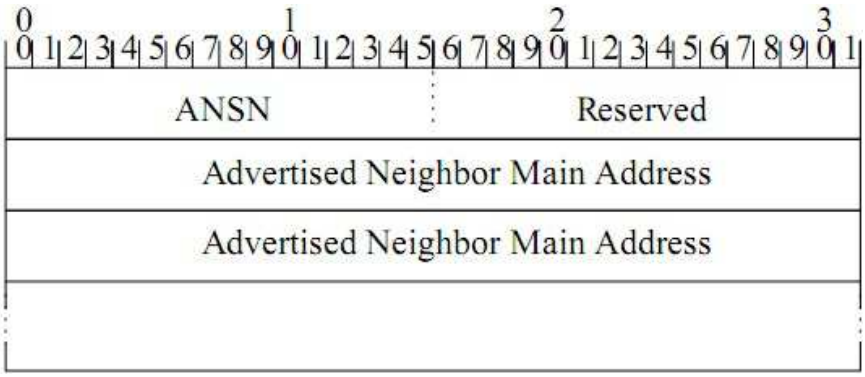

Figura 2.7: Mensagem TC.

Esta mensagem é enviada como parte da informação do formato geral de mensagens quando o campo Message Type é TC MESSAGE. O campo Time to Live deve ser 255 para que a mensagem seja difundida por toda rede. O campo Vtime é ajustado de acordo com o valor de TOP HOLD TIME. Os campos da mensagem TC estão descritos a seguir:

– **Advertised Neighbor Sequence Number (ANSN)**

  É um número de seqüência associado à configuração do vizinho anunciado, toda vez que o nodo detecta uma mudança nessa configuração, incrementa-se o número. Esse número é enviado no campo ANSN da mensagem TC para ficar atualizado com as mais novas informações. Quando um nodo recebe a mensagem TC, ele pode comparar se a informação recebida sobre os vizinhos anunciados é mais recente do que a que ele possui.

– **Advertised Neighbor Main Address**

  Este campo contém o endereço principal de um nodo vizinho. Todos os endereços principais dos vizinhos anunciados pelo nodo remetente são colocados na mensagem TC. Podem-se colocar endereços principais dos vizinhos anunciados extras para fins de redundância.

– **Reserved**

  Esse campo é reservado e dever ser configurado como "0000000000000000"para compatibilidade.

O formato do pacote OLSR para a mensagem TC pode ser visualizado na figura 2.8:

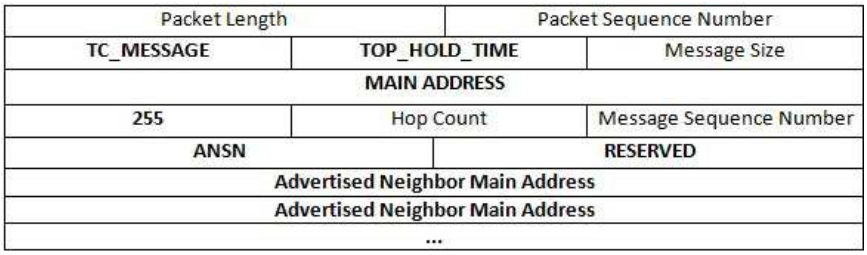

Figura 2.8: Formato do pacote OLSR com a mensagem TC.

Os campos TTL e Hop Count são os únicos mutáveis, ou seja, a mensagem TC sai pronta do nodo de origem e percorre toda a rede, sem modificações.

Uma mensagem TC é enviada por um nodo para rede para declarar uma configuração de enlaces, chamado de configuração de enlace anunciado, que deve incluir no mínimo os enlaces para todos os nodos que estão no seu Conjunto de Seletores MPRs. Isso faz com que o nodo não envie todos os seus enlaces diminuindo o tamanho das mensagens TC e somente os MPRs irão gerar mensagens TC diminuindo o tráfego de controle na rede.

Para construir o repositório Base de Informações Topológicas, cada nodo, que foi escolhido como MPR deve disseminar por toda rede as mensagens TC. As informações disseminadas pelas mensagens TC, ajudaram no cálculo das rotas de cada nodo. Quando uma mensagem TC é recebida o seguinte processo é executado:

– Se nenhuma entrada está registrada no repositório Base de Informações Topológicas com o endereço do remetente, é criada uma entrada com informações dos campos Vtime e ANSN de acordo com o cabeçalho da mensagem TC.

– Se nenhuma entrada está registrada no repositório Base de Informações Topológicas com o endereço do remetente e o valor de ANSN menor que o valor ANSN recebido, então essa entrada é atualizada

de acordo com as informações da mensagem TC.

– Se nenhuma entrada está registrada no repositório Base de Informações Topológicas com o endereço do remetente com valor de ANSN igual ao valor recebido, então o Vtime da entrada é atualizado.

2.1.9 Cálculo de rota

O cálculo de rota proposto na RFC 3626 consiste no SPF (Shortest Path First), o processo é elencado abaixo:

1. Adiciona-se todos os vizinhos simétricos de um salto na tabela de roteamento com Hop Count de 1;
2. Para cada vizinho simétrico de um salto, adicionam-se todos os vizinhos de dois saltos que estão registrados nesse vizinho que tenham:
   – Nenhum registro na tabela de roteamento.
   – Um enlace simétrico com o vizinho.

   Essas entradas são adicionadas com o Hop Count h valendo dois e próximo salto igual ao do vizinho.
3. Para cada nodo N da tabela de roteamento com Hop Count h igual a dois adiciona-se todas as entradas do repositório Base de Informações Topológicas onde:
   – O remetente tem a entrada da Base de Informações Topológicas igual ao de N.
   – O destinatário ainda não foi adicionado na tabela de roteamento.

   Essas entradas terão Hop Count igual a N+1 e próximo salto igual ao próximo salto do nodo que gerou a entrada.
4. Incrementa-se de um o Hop Count e repete-se o passo 3 até que todos os nodos com Hop Count igual ao novo h tenham sido processados.
5. Por último, a Base de informações sobre Associações de Múltiplas Interfaces é consultada. Caso a entrada tiver mais endereços de interfaces associados, é criada para cada associação uma nova tabela de roteamento, com h e próximo salto iguais ao valor original.

A tabela de roteamento é sempre atualizada quando acontecem mudanças nos seguintes repositórios:

– **Conjunto de Enlaces;**

– **Conjunto de Vizinhos;**

– **Conjunto de Vizinhos a dois Saltos;**

– **Base de Informações Topológicas;**

– **Base de informações sobre Associações de Múltiplas Interfaces.**

2.1.10 Visão Geral da Funcionalidade Núcleo

Nota-se que a funcionalidade núcleo é responsável pelo funcionamento básico utilizando OLSR e que os repositórios de informações são fundamentais para gerar o tráfego de controle. Baseado nas mensagens de controle os repositórios são dinamicamente atualizados. Na figura 2.9 pode-se ver a relação entre os repositórios de informações e a geração de mensagens, processamento de mensagens e cálculo de rota. Quando uma mensagem HELLO é recebida desencadeia atualizações no Conjunto de Enlaces, no Conjunto de Vizinhos a dois Saltos e no Conjunto de Seletores MPR. Mudanças no Conjunto de Enlaces provocam no Conjunto de Vizinhos e assim o cálculo do Conjunto de MPRs é refeito. Atualizações no Conjunto de Vizinhos a dois Saltos atualizam o Conjunto de MPRs. Quando uma mensagem TC é recebida a Base de Informações Topológicas sofre mudanças, assim acontece com a Base de Informações sobre Associações de Múltiplas Interfaces quando uma mensagem MID é recebida. A tabela de roteamento é construída com base no Conjunto de Vizinhos, Conjunto de Vizinhos a dois Saltos, Base de Informações Topológicas e a Base de Informações sobre Associações de Múltiplas Interfaces. Todas as mensagens inéditas são registradas na Base de Informações de Duplicatas que é consultado juntamente com o Conjunto de MPRs para determinar o tráfego de controle.

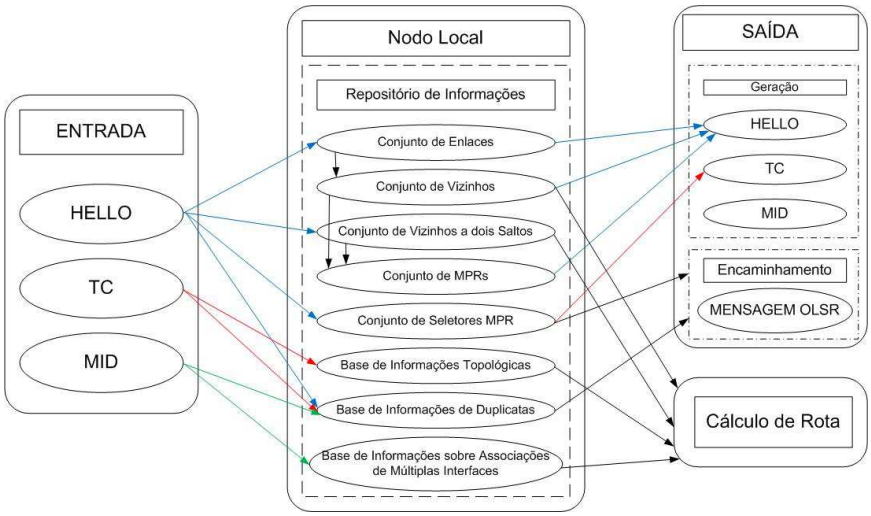

Figura 2.9: Relações entre os repositórios de informações e a funcionalidade Núcleo.

2.2 FUNÇÕES AUXILIARES

A RFC 3626 divide o protocolo OLSR em Funcionalidade Núcleo que já foi visto e Funções Auxiliares. As Funções Auxiliares provêm funcionalidades adicionais que podem ser empregadas em determinados cenários. Algumas funções auxiliares estão especificadas na RFC 3626, porém, somente a função HNA será descrita por ser a única relevante para este trabalho.

2.2.1 HNA

Uma rede Ad hoc pode existir isolada de outras redes, porém, existe a possibilidade de se comunicar com outras redes e ter acesso a internet. OLSR oferece este tipo de conectividade externa em nível de protocolo de roteamento. Um nodo anuncia-se como um gateway para uma determinada rede utilizando a mensagem HNA - Host and Network Association. Graças ao algoritmo padrão de encaminhamento, não é necessário que todos os nodos suportem a funcionalidade HNA para que as mensagens sejam inundadas (flooding) pela rede, porém, todos os nodos devem suportar o processamento do HNA caso se deseje trafegar pacotes advindos de outras redes. Na figura 2.10 ilustra um caso típico onde se utiliza HNA.

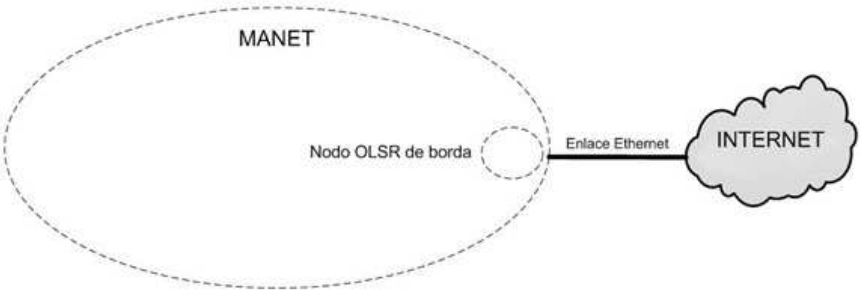

Figura 2.10: Cenário HNA.

O formato da mensagem HNA é basicamente uma lista de endereços de rede e máscaras de rede, pode ser visto na figura2.11.

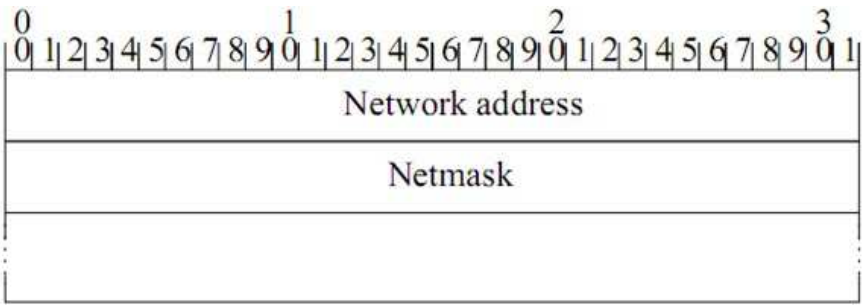

Figura 2.11: Formato da Mensagem HNA.

Esta mensagem é enviada como parte do pacote OLSR com o campo Message Type sendo HNA MESSAGE e o campo TTL deve ser 255. Nenhum campo da mensagem HNA é modificado durante o encaminhamento da mensagem.

Cada nodo mantém informações sobre quais nodos atuam como gateway atualizando o repositório Base de Informações HNA. Cada entrada HNA contém as seguintes informações:

- Gateway: endereço principal do nodo que tenha anunciado conectividade com outra rede, este valor é capturado do campo Originator Address.
- Endereço de Rede: Rede a ser comunicada
- Máscara de Rede: Máscara da rede a ser comunicada
- Vtime: Tempo de validade da entrada, capturada do campo Vtime.

A cada mudança no repositório Base de Informações HNA ou na topologia de rede as rotas HNA devem ser calculadas. Caso exista mais de um gateway para mesma rede, o gateway mais próximo em Hop Count é escolhido. O roteamento no HNA é feito salto a salto. Isto significa que as rotas HNA são adicionadas com o próximo salto na rota para o Gateway. O verdadeiro Gateway que anunciou conectividade com outra rede, não é adicionado na tabela de roteamento de um nodo intermediário, será adicionado como gateway o nodo de próximo salto até chegar no verdadeiro Gateway.

2.3 FORMATO GERAL DE PACOTE/MENSAGEM PARA MANET CONFORME RFC 5444

A RFC 5444[6] especifica um formato de pacote projetado para transportar múltiplas mensagens dos protocolos de roteamento para troca de informações entre roteadores MANET. A mensagem consiste em um cabeçalho, para o controle da disseminação da mensagem, e um corpo da mensagem, onde contém informações do protocolo. A especificação foi projetada para que o protocolo seja:

- Extensível a novas mensagens e novas estruturas TLVs (Type-Length-Value), que é uma forma genérica de representar um atributo;
- Eficiente em relação à redução do número de octetos e pacotes transmitidos, utilizando estruturas de pacote que comportam múltiplas mensagens;

– Capaz de separar encaminhamento do processamento, ou seja, o encaminhamento é feito utilizando as informações do cabeçalho, tornando o processamento do corpo da mensagem desnecessário.

2.3.1 Formato de mensagem

Pacotes podem conter uma ou mais mensagens. As mensagens contêm:

– Um cabeçalho da mensagem

– Um bloco TLV que pode conter zero ou mais TLVs, associado a toda mensagem.

– Zero ou mais blocos de endereço, cada um contendo um ou mais entradas.

– Um bloco de endereço bloco TLV, contendo zero ou mais TLVs e seguindo cada bloco de endereço, o qual cada endereço pode ser associado com um atributo adicional.

A mensagem é definida na tabela 2.4 a seguir:

MENSAGEM	MSG-HEADER
	TLV-BLOCK
	ADDR-BLOCK/TLV-BLOCK
Cabeçalho da Mensagem	MSG-TYPE
	MSG-FLAG
	MSG-ADDR-LENGTH
	MSG-SIZE
	MSG-ORIG-ADDR
	MSG-HOP-LIMIT
	MSG-HOP-COUNT
	MSG-SEQ-NUM

Tabela 2.4: Formato da Mensagem.

Onde:

– TLV-BLOCK é um grupo de TLVs

– ADDR-BLOCK especifica um ou mais endereços

– MSG-TYPE especifica o tipo de mensagem

– MSG-FLAG especifica a interpretação do restante do cabeçalho, podendo ser bit 0, bit 1, bit2 ou bit3

– MSG-ADDR-LENGTH codifica o tamanho de todos os endereços da mensagem.

– MSG-SIZE especifica o número de octetos da mensagem

– MSG-ORIG-ADDR identifica o roteador MANET que originou a mensagem

– MSG-HOP-LIMIT é o número máximo de saltos que a mensagem deve ser transmitida

– MSG-HOP-COUNT é o número de saltos que a mensagem já percorreu

– MSG-SEQ-NUM é o número de seqüência gerado pelo roteador que originou a mensagem

3 MAE PARA OLSR

3.1 SEGURANÇA PARA OLSR

O protocolo OLSR,assim como outros protocolos de roteamento para redes Ad Hoc, apresentam algumas vulnerabilidades[3], como a modificação, personificação e fabricação de mensagens. Foram apresentadas algumas propostas de segurança, como em *Secure Extension to the OLSR protocol*[11], e em *UM MODELO DE SEGURANÇA PARA REDES MÓVEIS AD HOC*[3], sendo que o primeiro é especifico para o protocolo e o segundo é extensível para qualquer protocolo para redes móveis. Uma das maneiras de se aplicar as propostas de segurança é o desenvolvimento de uma extensão para as implementações existentes, que no caso do OLSRD poderá ser feito por meio de um plugin. Neste trabalho foi elaborado um plugin que implementa a Extensão de Autenticação para MANET[3].

3.1.1 Vulnerabilidades do protocolo OLSR

Pode-se enxergar o protocolo OLSR com dois grupos de mensagens. O primeiro grupo contém a mensagem do tipo HELLO que não é encaminhada pelos nodos, ou seja, tem apenas um salto. O segundo grupo contém mensagens do tipo TC, MID e HNA que são encaminhadas por toda rede. Portanto, ataques de modificação de mensagens não se aplicam ao primeiro grupo, apenas para mensagens TC, MID e HNA. Além disso, os campos dessas mensagens não são modificados durante o encaminhamento das mesmas, ou seja, a mensagem chega ao seu destino da mesma forma que saiu, tornando também todo ataque de modificação em ataque de personificação. Os ataques de fabricação são possíveis em ambos os grupos de mensagens e ainda podem ser combinados com ataques de personificação. Essas vulnerabilidades são consideradas para a especificação original[4] e podem ser corrigidas adicionando uma MAE adequada para o protocolo de roteamento[3]. Para ilustrar os ataques segue a tabela 3.1.

Ataque	Grupo de Mensagem	Alvo	Informação de Origem na Mensagem Corrompida
Fabricação	Grupo 1	Conjunto de Vizinhos	Qualquer
Fabricação + Personificação	Grupo 1	Status do Enlace	Endereço IP do nodo personificado
Fabricação	Grupo 2	Conjunto Seletores MPR	Qualquer
Modificação + Personificação	Grupo 2	Número de Seqüência	Endereço IP do remetente

Tabela 3.1: Ataques no OLSR.

Fabricação de mensagens Grupo 1 (HELLO): A figura 3.1 ilustra este ataque onde o adversário gera mensagens HELLO falsas tornando o enlace simétrico. Desta forma, todos os vizinhos do adversário escolhem-no como único MPR. Agora todo tráfego com destino aos nodos que não são vizinhos diretos de um desses nodos, vai ser encaminhado para o adversário.

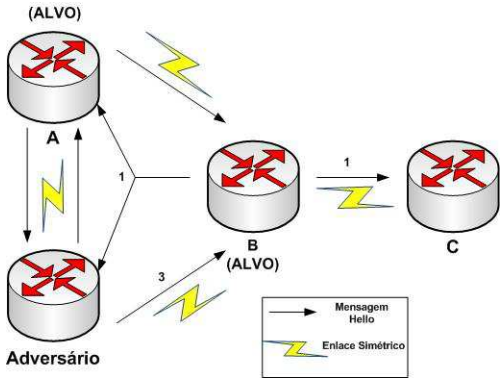

Figura 3.1: Fabricação de mensagem HELLO.

Cenário anterior ao ataque: A (ALVO) tem B(ALVO) como MPR, encaminhando o tráfego de A para C. C está a dois saltos de A e do adversário.

1- Mensagem HELLO de B: Adversário verifica que A e C são vizinhos de B.

2- Mensagem HELLO de A: Adversário verifica que B é vizinho direto de A.

3- Ao receber mensagem HELLO de B, o adversário fabrica uma falsa mensagem HELLO anunciando A, B, C e Z (endereço ainda não utilizado) com status de enlace simétrico. O nodos A e B selecionam o adversário como novo MPR, impedindo o tráfego de A para C.

Fabricação e Personificação de mensagens Grupo 1 (HELLO): A figura 3.2 mostra o ataque onde um adversário fabrica uma mensagem HELLO personificada.

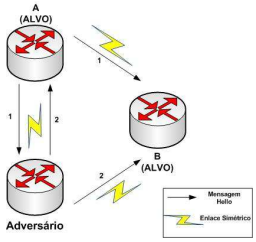

Figura 3.2: Fabricação e Personificação de mensagem HELLO.

Cenário anterior ao ataque: A (ALVO) e B(ALVO) têm um enlace simétrico entre si adquirido pela detecção de enlace simétrico ilustrado na figura 2.6.

1 - Mensagem HELLO de A: Adversário verifica que B é vizinho direto de A.

2 - Após processar a mensagem HELLO de A, o adversário fabrica uma mensagem HELLO personificando A e anuncia B com status de enlace perdido. Isso faz com que B torne o estado do enlace para assimétrico, impedindo o tráfego pelo enlace.

Fabricação de Mensagens Grupo 2: Na figura 3.3 ilustra o ataque onde um adversário fabrica uma mensagem TC anunciando vizinhos a dois ou mais saltos como parte do seu Conjunto Seletor MPR.

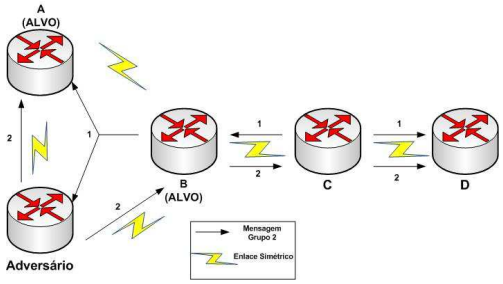

Figura 3.3: Fabricação de mensagem do Grupo 2.

Cenário anterior ao ataque: A(ALVO) apresenta uma rota para D através do roteador B(ALVO).

1 - Mensagem TC de C: Adversário verifica que D está a três saltos.

2 - Após processar a mensagem TC, o adversário fabrica uma mensagem TC, anunciando D como seu vizinho direto. Agora A passa a trafegar dados para D através do Adversário.

O mesmo raciocínio pode ser utilizado para mensagens MID e HNA, porém, com intuitos diferentes.

Modificação e Personificação de mensagens Grupo 2: Na figura 3.4 ilustra o ataque onde um adversário altera o campo **Message Sequence Number** de uma mensagem TC antes de envia-lá.

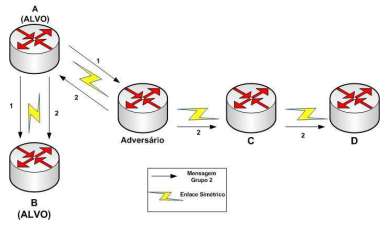

Figura 3.4: Modificação e Personificação de Mensagens TC.

1 - Mensagem TC de A(ALVO): "Message Sequence Number"é X.

2 - O adversário encaminha a mensagem TC com o valor do Message Sequence Number igual a X+n (valor inteiro qualquer), agora não haverá encaminhamento de mensagens TC de A para C e D até que o valor Message Sequence Number for menor que X+n.

3.1.2 MAE para OLSR

Em *UM MODELO DE SEGURANÇA PARA REDES MÓVEIS AD HOC*[3], foi proposto uma Extensão de Autenticação para Manet(MAE). Ela consiste em um protocolo de autenticação de mensagens de aplicações onde não há uma hierarquia cliente-servidor. Um exemplo é o protocolo OLSR. A MAE consiste em objetos de autenticação que possuem os campos essenciais de uma mensagem em que se deseja autenticar, assegurando a autenticidade e integridade dessas, há sempre um objeto de autenticação obrigatório, que pode ser uma assinatura digital ou um código de autenticação de mensagem. Para o protocolo OLSR foi definido que esse objeto de autenticação obrigatório será uma assinatura digital, em que consiste em um *hash* da mensagem com a chave privada do host de origem. Essa operação, junto com um certificado válido, garante a origem da mensagem. Esse tipo de abordagem é utilizado em ambientes que utilizam criptografia assimétrica, ou seja, não há um compartilhamento prévio das chaves. A assinatura digital é utilizado nos campos não mutáveis das mensagens. Os campos hop count e TTL das mensagens MID, HNA e TC são mutáveis, portanto precisam ser autenticados por outro mecanismo diferente da assinatura digital. O mecanismo adotado pela MAE é o da Cadeia de Hash[3],porém, não foi implementado nesse trabalho e está tratado como trabalhos futuros para aprimorar o plugin MAE com uma função que implementa a cadeia de hash para autenticar campos da mensagem mutáveis como TTL e Hop Count. Um campo opcional da MAE são os Certificados(CERT). É possível enviar, junto com uma assinatura digital, um certificado para garantir a confiabilidade do signatário.

3.1.3 Formato das mensagens MAE

Para atender o formato de mensagem descrito em [6], as mensagens MAE seguirão o seguinte formato:

Figura 3.5: Mensagem MAE genérica.

– MSG TYPE: depende do tipo da mensagem MAE, pode ser HASH, AUTH ou CERT.

– FLAGS: deve ser setado como zero

– ADDR LENGTH: deve ser setado como zero

– MSG SIZE: cabeçalho mais o tamanho do objeto de autenticação

– HOP LIMIT: deve ser setado como zero

– HOP COUNT: deve ser setado como zero

– MSG SEQ NUMBER: deve ser setado como zero

– TLV BLOCK: deve estar o objeto de autenticação que pode ser DS, MAC, CERT ou HASH.

As mensagens MAE estão descritas abaixo.

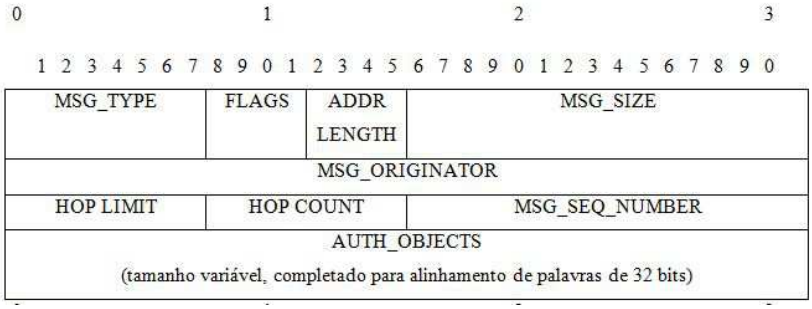

Figura 3.6: Cabeçalho do Auth.

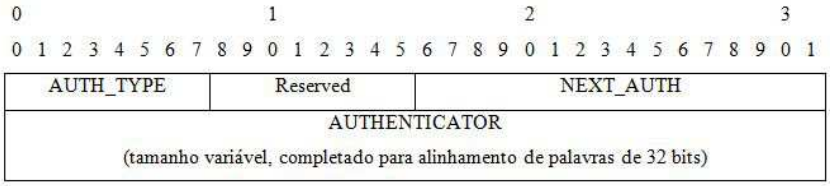

Figura 3.7: Objeto de Autenticação.

Figura 3.8: Cabeçalho do Cert.

Figura 3.9: Objeto do certificado.

3.1.4 Certificação em MANET

Para assegurar a confiabilidade dos nodos um modelo de certificação deve ser elaborado, seguindo os princípios básicos:

- Emissão e renovação de certificados: caso um nodo não possua certificado ou este expirou, ele deve solicitar aos nodos que possuam uma estrutura de certificação para que se emita ou renove o certificado. Deve-se ter uma estrutura que aceite ou negue a renovação e emissão de certificados.
- Revogação do certificado: caso um nodo seja comprometido, a estrutura de certificação deverá ser capaz de identificar e revogar o certificado desse nodo.
- Validação do certificado: cada nodo deverá ser capaz de identificar se o certificado é válido ou não, de acordo com as informações extraídas desse certificado, como tempo de expiração e data de emissão.

Um mecanismo de certificação é necessário para um funcionamento mais dinâmico do serviço de autenticação, um projeto é previsto em [3]. O trabalho supõe que os certificados foram distribuídos anteriormente, e sempre que um pacote é enviado, uma mensagem com o certificado do nodo é enviada.

3.1.5 Sugestão de autenticação e integridade proposta pela RFC 5444

A RFC 5444 não descreve um protocolo e sim um formato de pacote, assim, não é especificada nenhuma consideração de segurança da informação. Todavia, alguns mecanismos de segurança são habilitados por essa especificação e podem fazer parte de um protocolo utilizando esta especificação. A RFC 5444 sugere a autenticação e integridade baseadas em:

- Mensagens são projetadas para serem transportadoras de informações do protocolo e podem, a cada salto, ser encaminhada e ou processadas pelo protocolo utilizando esta especificação.
- Pacotes foram projetados para transportar um número de mensagens entre roteadores MANETs vizinhos em uma única transmissão e sobre um único salto lógico.

Conseqüentemente:

– Para mensagens encaminhadas onde a mensagem é imutável pelos roteadores intermediários que a encaminham, autenticação e integridade pode ser implementada fim a fim entre os roteadores incluindo uma mensagem TLV adequada contendo uma assinatura criptográfica na mensagem. Os campos MSG-HOP-COUNT e MSG-HOP-LIMIT são os únicos que sofrem modificações quando a mensagem é encaminhada, assim, essa assinatura pode ser baseada na mensagem inteira, zerando os dois campos citados acima.

– Autenticação e integridade deve ser implementada salto a salto em nível de pacote entre roteadores, incluindo um pacote TLV adequado contendo uma assinatura criptográfica ao pacote. Essa assinatura pode ser calculada baseada no pacote inteiro ou uma parte.

A extensão de autenticação para MANET proposta neste trabalho utiliza exatamente os conceitos sugeridos pela RFC 5444 utilizando o protocolo de roteamento OLSR com formatação de mensagem conforme a RFC 5444.

4 IMPLEMENTAÇÃO

4.1 INTRODUÇÃO

Para a consolidação da RFC 3626 [4], descrita no capítulo anterior, foi necessário implementar um programa que atende as especificações dessa RFC. Algumas implementações surgiram, dentro as quais a que mais se destacou foi a Unik OLSR Daemon[5]. Essa implementação está tendo uma continuidade, com novas versões surgindo a cada momento. Comumente ela é chamada de OLSR daemon ou simplesmente OLSRD. Uma das vantagens do OLSRD[5] foi o fato dela atender a RFC 3626 de maneira fiel, inclusive com os campos opcionais. O OLSR daemon possui uma estrutura que se permite a introdução de plugins permitindo extensões das funcionalidades do protocolo. Pela sua popularidade, arquitetura e possibilidade de novas extensões, foi utilizado o OLSRD como base deste trabalho. O capítulo descreverá as principais funcionalidades, como a geração de pacotes, o arquivo de configuração e a interface plugin - OLSRD.

4.2 OLSRD

Desenvolvido originalmente para GNU/Linux em C, o OLSRD já pode ser encontrado para diversas plataformas como Windows, Mac OS, smartphones etc. Ele tem suporte a bibliotecas dinâmicas (DLL) e uma versão GUI é disponível. O desenvolvimento seguiu os princípios abaixo devido sua extensão e complexidade:

- **Modularidade:** Qualquer código que possa ser entendido como um mecanismo geral e que possa ser utilizado por qualquer entidade tem de ser o mais modular possível. O que significa que as entidades que utilizam tais funções devem se registrar dinamicamente na função[5].
- **Estrutura de dados consistente:** por se tratar de um protocolo baseado em tabelas, todos os dados contidos têm de ser consistentes e as tabelas devem possuir a mesma estrutura.
- **Transparência IP:** O deamon tem de operar tanto para IPv6 quanto para IPv4, portanto todas as operações tem de ser compatíveis com o endereçamento de 32bits do IPv4[8] e 128 do IPv6[9].
- **Código legível:** o código tem de ser de fácil leitura para um observador.

– **Código independente de Plataforma:** o código dependente da plataforma tem de ser separado do resto do código de maneira modular, assim facilita a implementação em outras plataformas.

A implementação do OLSRD[5] define as seguintes entidades:

– **Socket Parser**;

– **Packet Parser**;

– **Repositórios de Informações**;

– **Scheduler**.

E pode ser visto na figura 4.1com mais detalhes.

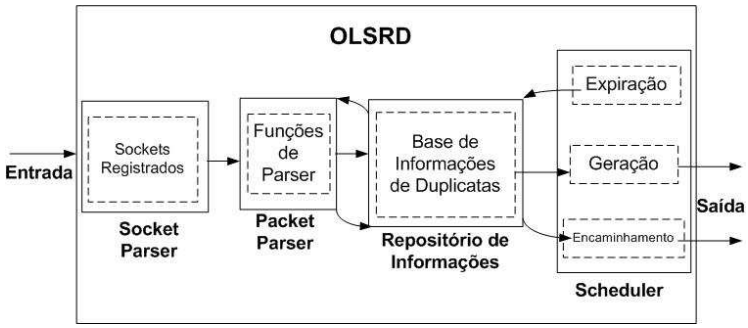

Figura 4.1: Estrutura de Funcionamento do OLSRD.

4.2.1 Socket Parser

Todo o tráfego OLSR primeiramente passa pelo Socket Parser. Esta entidade tem a responsabilidade de escutar toda a informação em um grupo de sockets. Os sockets e suas funções podem ser registrados a todo instante. O Socket Parser checa todo o tráfego utilizando um loop, chama a função associada ao socket que tem o dado que está chegando. Esta funcionalidade permite que múltiplas entidades escutem múltiplos tipos de informação. O esquema do funcionamento do Socket Parser está na figura 4.2.

4.2.2 Packet Parser

Ao iniciar o OLSRD registra todos os sockets do tráfego de controle com o Socket Parser. Esses sockets são registrados com uma função de parser que é chamada sempre que a informação está disponível. O Packet Parser recebe todo o tráfego OLSR que foi enviado via broadcast na porta UDP 698 e checa se o

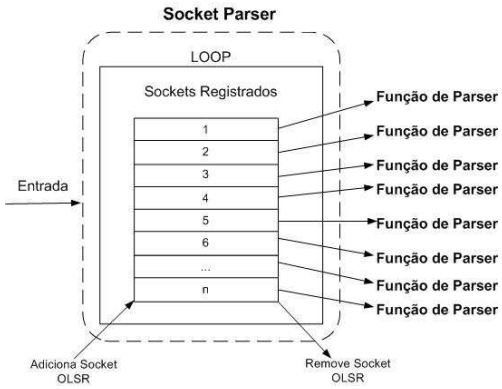

Figura 4.2: Esquema do Socket Parser.

tamanho reportado no cabeçalho OLSR corresponde com a quantidade de informação recebida. Caso positivo, o pacote é transformado em mensagens e as mesmas são registradas com funções parser de mensagens em seus nomes. Caso negativo, o pacote é silenciosamente descartado. O Packet Parser foi projetado de maneira similar ao Socket Parser. Funções parser de mensagens podem ser registradas e removidas para qualquer tipo de mensagem dinamicamente. Após o processamento das mensagens as mesmas são enviadas pelo algoritmo padrão de encaminhamento. A figura 4.3 ilustra o esquema do Packet Parser.

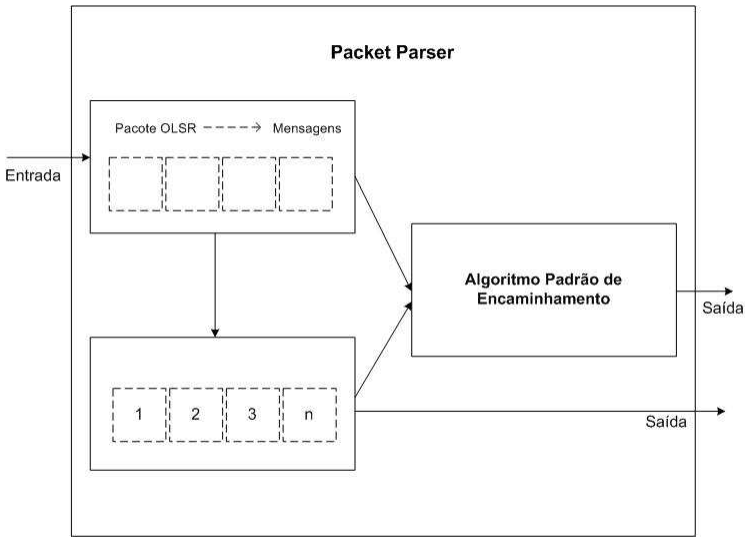

Figura 4.3: Esquema do Packet Parser.

4.2.3 Repositório de Informações

OLSR é um protocolo de roteamento baseado em tabelas. Isso significa que a informação é atualizada e removida dinamicamente com mudanças na topologia da rede dentre outros fatores. As informações são armazenadas em tabelas que devem ser projetadas de maneira inteligente, pois as tabelas são consultadas constantemente. OLSRD implementa os repositórios de informações por listas encadeadas e indexadas por

hashes. As listas encadeadas são capazes de estruturar conjunto de dados de tamanhos dinâmicos. Uma lista encadeada é apenas uma lista de entradas onde cada entrada faz referência para a próxima. A última entrada pode fazer referência a primeira entrada, tornado o processo em um ciclo. Podem-se introduzir listas encadeadas duplas, onde todas as entradas mantêm para a entrada anterior e a próxima, deste modo, não é necessário o conhecimento de outros elementos da lista além dos elementos atuais para remoção de algum elemento. Além disso, uma lista encadeada dupla pode ser orientada para diminuir o tempo de procura, executando a procura na lista para trás ou para frente. Um hash é a transformação de um determinado valor original, em um valor ou chave de tamanho fixo geralmente menor que representa o valor original. No contexto apresentado, o hash é utilizado para indexar e recuperar itens em banco de dados e cadeias de hash, onde as entradas foram indexadas por um hash. Uma função de hash é utilizada para geras os hashes, que devem ser únicos e irreversíveis. É desejável que se utilize funções de hash que propiciam menor probabilidade de colisão, ou seja, criação de hashes iguais a partir de sementes diferentes. Além disso, uma função de hash deve prover sempre o mesmo hash para uma determinada semente. Em estrutura de dados os hashes são chaves que indexam itens, facilitando o processo de busca e atualização dos dados. OLSR não foi projetado para grandes redes, portanto, as tabelas no OLSR comportam uma quantidade de entrada relativamente pequena. Assim, as tabelas no OLSRD geralmente registram uma entrada por nodo, tornando possível a utilização de bases de dados com listas encadeadas duplas indexadas por hashes[7]. OLSRD utiliza um hash baseado no endereço IP principal de um nodo para indexar o nodo em um vetor estaticamente alocado. Cada elemento desse vetor é um elemento raiz na lista encadeada dupla. Esta estrutura pode ser vista na figura 4.4. Os elementos na lista não estão ordenados e o seu tamanho tão grande quanto ao maior hash possível derivado de um endereço IP.

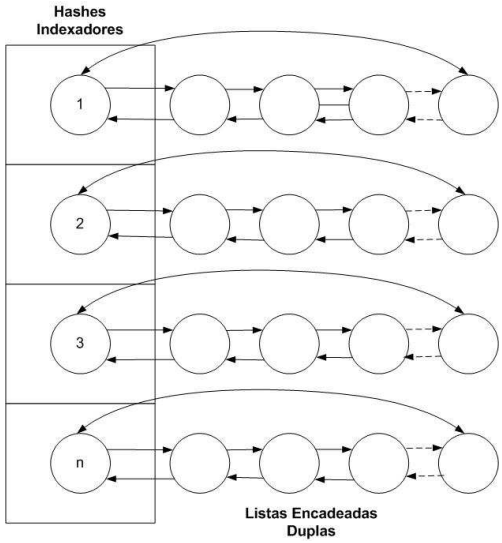

Figura 4.4: Indexação de informação no OLSRD.

4.2.4 Scheduler

O **Scheduler** é um agendador do OLSRD que realiza tarefas em intervalos definidos. Esse intervalo deve comportar o intervalo de tempo o qual todas as tarefas regulares sejam executadas. Se certa entidade deseja que alguma tarefa seja realizada em intervalos regulares, deve-se registrar uma função no agendador para tal. Ocorre uma verificação nos temporizadores de cada função registrada em cada ciclo. Isto é feito para determinar se uma tarefa vai ser executada ou não. Quando um temporizador está expirado na próxima chamada o mesmo é reiniciado. Posteriormente, as funções registradas que devem ser executadas são chamadas e mudanças de topologia e vizinhança são processadas. O tempo gasto para executar todo esse processo é o tempo de um ciclo.

4.3 FUNCIONAMENTO DO PLUGIN

O OLSRD permite que se faça extensões do funcionamento OLSR por meio de plugins. O plugin pode adicionar ou mesmo alterar funções que foram implementadas pelo OLSRD. Pelo fato do código ser modular, o plugin tem acesso a quase todas as funções da implementação, o que o torna extremamente versátil.

4.3.1 Olsrd plugins

As principais vantagens de se ter a possibilidade de plugins são:

- Não há necessidade de se modificar o código original.
- O plugin poderá ser licenciado e distribuído de acordo com uma política própria. O OLSRD foi licenciado para o BSD, qualquer modificação no código original teria de seguir a licença original.
- Há a possibilidade de escrever o plugin em qualquer linguagem que implemente biblioteca dinâmica
- Para fins de compatibilidade nenhuma ou pouca modificação no plugin será necessária com novas versões

O OLSR provê um algoritmo padrão de encaminhamento que pode ser utilizado por mensagens de tipo desconhecidas. Isso é útil para a transmissão de mensagens particulares, que podem ser produzidas por meio de um plugin. O plugin mostra a força da implementação modular, já que ele pode registrar funções

no Socket dinamicamente, podendo modificar pacotes que seriam enviados e pode tratar dados de um determinado tipo.

4.3.2 A interface Plugin

O plugin pode ser descrito como na figura 4.5:

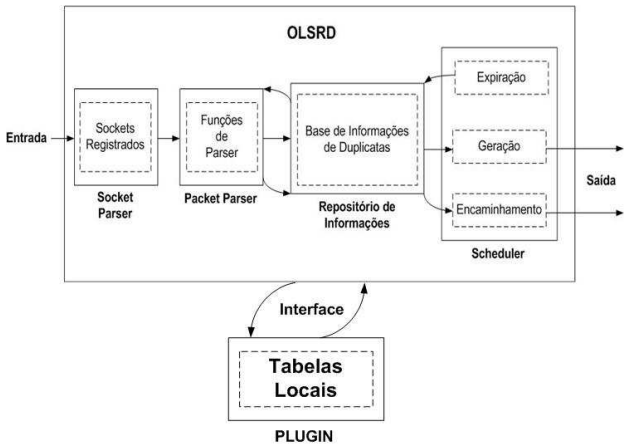

Figura 4.5: Esquema do Plugin.

Ele tem acesso as principais funções do deamon, e poderá realizar as seguintes operações:

- Socket Parser: o plugin pode tanto registrar quanto remover sockets
- Packet Parser: pode-se adicionar funções que alterem ou tratem diferentes tipos de mensagens.Útil para o caso em que se introduza novas mensagens ou que se deseje modificar uma mensagem recebida
- Repositório de Informações: pode-se ter acesso a todas as tabelas do OLSRD, onde se pode, inclusive, modificar os dados.
- Scheduler: esse módulo pode-se determinar que um certo evento, como enviou de um tipo de mensagem, ocorra de tempos em tempos. Isso é útil para a elaboração de novas mensagems, no agendador pode-se ter acesso as funções de transmissão do OLSRD.

A comunicação do plugin com o OLSRD é ilustrado na figura 4.6.

A interface entre o plugin e o daemon permite que o plugin possa sempre saber o que esperar do daemon e o deamon do plugin. Assim sempre que ocorrer um determinado evento o daemon chamará o plugin para buscar uma variável ou para executar uma determinada função.

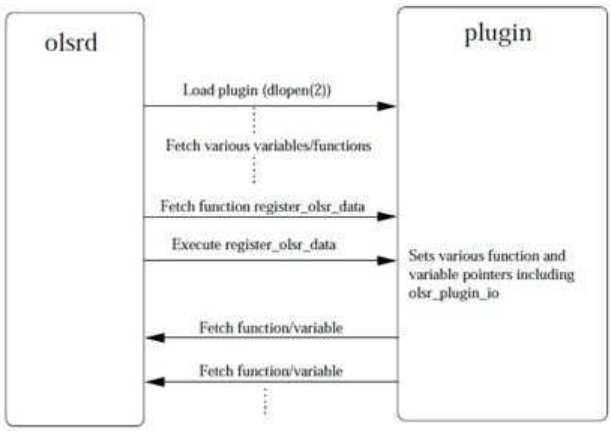

Figura 4.6: Processo de Inicialização do Plugin.

4.4 IMPLEMENTAÇÃO PLUGIN MAE PARA OLSR

A fim de se implementar a MAE[3] para o OLSRD, foi utilizada o modelo de plugin apresentado em [5]. O plugin será um intermediário entre o OLSRD e o MAE e pode ser representado pela figura 4.7

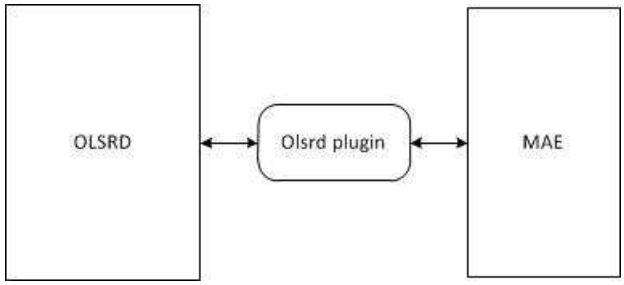

Figura 4.7: OLSRD/MAE.

Onde o plugin é a interface entre o deamon e a Extensão de Autenticação para Manet[Puttini]. Basicamente o olsrd plugin utilizará os dados do OLSRD a fim de se aplicar as funções da MAE. O código está disponível com licença GPL em `http://maepluginolsrd.sourceforge.net/`(olsrd mae plugin).

4.4.1 Funções MAE

As funções relativas ao MAE estão no arquivo mae.c e são as seguintes:

– *setup cert header*: define uma mensagem CERT com base em um certificado. Nessa função verifica-se o tipo de certificado e armazena as chaves que serão utilizadas para a função de autenticação. Essa função chama a *verify certificate* para validar o certificado.A modificação do trabalho foi adaptar a mensagem CERT à RFC 5444.

– *Verify certificate*: recebe como entrada o certificado e verifica-se se o mesmo é válido.

– *Setup hash header*: define uma mensagem HASH. No trabalho apenas o cabeçalho foi incluído, deve-se implementar em um trabalho futuro o objeto de autenticação Cadeia de Hash. A mensagem segue o formato RFC 5444

– *Setup auth header*: define uma mensagem AUTH.Essa função apenas preenche o cabeçalho das mensagens AUTH com base no certificado presente no nodo. O trabalho adaptou o preenchimento da mensagem para a RFC 5444

– *Set authenticator*: executa a função de assinar a mensagem. Recebe uma mensagem e com a chave privada, obtida anteriormente, assina, utilizando funções do OpenSSL.

– *Verify authenticator*: verifica se a assinatura que foi feita é válida com base em funções do OpenSSL.

4.4.2 O Plugin OLSRD MAE

A MAE para o OLSRD precisa realizar as seguintes ações:

– Verificar se o nodo possui um certificado válido para autenticação das mensagens;

– Assinar as mensagens do tipo HELLO, HNA, MID e TC;

– Implementar a Cadeia de Hash para as mensagens TC, MID e HNA;

– Criar mensagens MAE com os objetos de autenticação de cada mensagem;

– Verificar se as mensagens recebidas estão corretamente assinadas.

Para que essas funções sejam implementadas, o plugin deverá:

– Abrir um arquivo especifico com o certificado válido do nodo;

– Utilizar uma função que analisa as mensagens antes de serem enviadas;

– Utilizar uma função que adiciona campos a uma mensagem;

– Analisar as mensagens que chegam no nodo;

– Aceitar ou descartar mensagens.

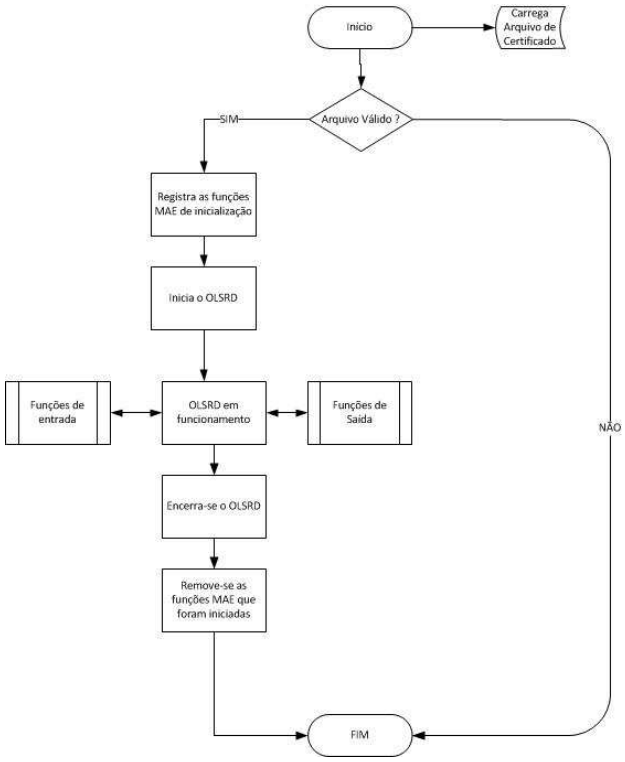

Figura 4.8: Fluxograma.

4.4.3 Arquivos OLSRD MAE Plugin

Os arquivos da implementação foram divididos em:

- Makefile: contêm as informações relativas à compilação.
- Src/(olsrd plugin.c): contêm as informações que serão lidas pelo arquivo de configuração do OLSR, como a localização do arquivo de certificado do nodo.
- Src/mae.h: contem as principais definições da MAE, como o formato das mensagens.
- Src/mae.c: contêm as funções da MAE.
- Src/(olsrd mae.h): contêm as definições da interface plugin/MAE.
- Src/(olsrd mae.c): contêm a implementação da interface plugin/MAE.

4.4.4 Parâmetros de configuração

Para que o OLSRD carregue o plugin é necessário adicionar alguns campos no arquivo de configuração que pode ser ilustrado por: load plugin *nome do plugin.versão* Os parâmetros que serão utilizados pelo plugin e que devem ser lidos pelo arquivo de configuração deverão estar no formato Plparameters

localização do arquivo de certificado Os parâmetros são lidos a partir da função *olsrd plugin parameters plugin parameters[]* do arquivo *olsrd plugin.c* e tratados de acordo com a especificação do plugin. No caso, o certificado será armazenado em uma variável global de nome *certfile*.

4.4.5 Funções do plugin

As funções do plugin podem ser divididas em três partes principais: funções de inicialização, funções de entrada e funções de saída. A seguir a descrição de cada uma.

4.4.5.1 Funções de inicialização

As funções de inicialização são aquelas que ocorrem no momento em que o deamon é acionado. Essas funções mostrarão para o OLSRD quais parâmetros devem ser alocados para o plugin. As funções de inicialização estão na função int *olsrd mae plugin init(void)*. As funções de inicialização são:

- *add ptf*: essa função aloca outra função para que se modifique todos pacotes OLSR antes dele ser enviado. Para o caso do plugin foi acionada a função *add auth*, que estão as funções de saída;
- *olsr preprocessor add function*: essa função aloca uma outra função para que se possa pré-processar um pacote recebido. A função *mae preprocessor* é a responsável por esse pré-processamento e a responsável pelas funções de entrada
- *setup cert header*: essa função é do arquivo mae.c e serve para que se defina a mensagem CERT, formato no capítulo 3 [mensagem cert], já que esta será a mesma para todas as mensagens do mesmo nodo.
- *setup auth header*: essa função define o cabeçalho das mensagens do tipo AUTH, que é definido no capítulo 3[mensagem auth]. Apenas o campo Authenticator da mensagem AUTH é modificado de acordo com a mensagem olsr.
- *setup hash header*: essa função define o cabeçalho das mensagens do tipo HASH, que é definido no capítulo 3[mensagem hash]. Apenas o campo *HASH CHAIN* da mensagem HASH é modificado de acordo com a mensagem olsr.

Caso não ocorra nenhum erro, retorna-se 1 e o plugin é inicializado. Na figura 4.9 ilustra o diagrama de funções de inicialização.

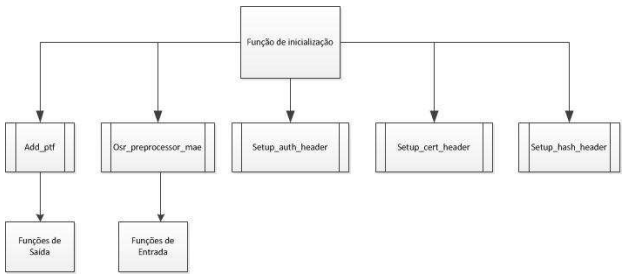

Figura 4.9: Diagrama de funções de inicialização.

4.4.5.2 Funções de Saída

Sempre que houver um pacote para ser enviado a função *add ptf* chamará a função *add auth* e esta poderá modificar o pacote. A função *add auth* realiza os seguintes procedimentos:

1. Copia o conteúdo da parte de dados do pacote para uma variável msg;
2. Chama a função *olsr set manet authentication extention*, que recebe os parâmetros msg e o tamanho do pacote. Essa última função, por sua vez, é responsável por autenticar as mensagens olsr que estão no pacote. Para tal ela segue os seguintes passos:
   - Segmenta a mensagem do pacote em diversas mensagens olsr;
   - Verifica o tipo de mensagem e adiciona uma mensagem de assinatura e de cadeia de hash se for o caso;
   - Adiciona uma mensagem de CERT no final de todas as mensagens.

O diagrama de funções de saída pode ser visto na figura 4.10.

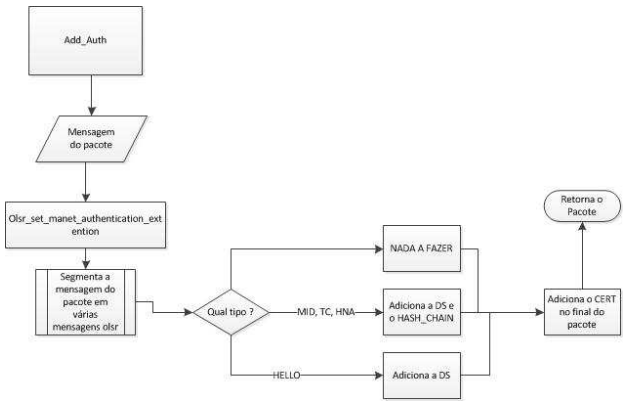

Figura 4.10: Diagrama de funções de saida.

O objeto de autenticação da mensagem AUTH é definido na função *olsr set authenticator*, onde se zera todos os campos mutáveis e assina toda a mensagem olsr, inclusive com os campos da mensagem AUTH.

A assinatura é feita pela função *set authenticator* que é definida no arquivo mae.c. O objeto de autenticação da mensagem HASH não foi concluído, apenas o cabeçalho.

4.4.5.3 Funções de Entrada

A função *mae preprocessor* recebe todos os pacotes que chegam no nodo e pode modificá-los antes que esse seja processado pelo daemon. Essa função realiza os seguintes procedimentos:

1. Copia o conteúdo do pacote para uma estrutura conhecida (OLSR);
2. Verifica se os dados do pacote são válidos, utilizando a função *olsr verify authentication()*. Se a função retornar 1, o pacote será devolvido para o daemon, senão será descartado silenciosamente. A função *olsr verify authentication()* realiza um procedimento similar ao do *olsr set manet authentication extention()*.
3. Segmenta a mensagem do pacote em diversas mensagens olsr e na sua mensagem AUTH correspondente;
4. Verifica se na mensagem CERT há um certificado válido;
5. Verifica se as mensagens AUTH correspondem a suas mensagens olsr.

O diagrama de funções de entrada pode ser visto na figura 4.11.

Caso algum mensagem não seja válida, todo o pacote será descartado.

4.5 VISÃO GERAL DA IMPLEMENTAÇÃO

Pelo fato das funções implementadas tratarem pacotes e não mensagens foi necessário fazer as seguintes considerações para a utilização da MAE no OLSRD.

– As mensagens MAE não fazem parte da mensagem OLSR, mas sim do pacote;

– Para cada mensagem OLSR dos tipos TC,HELLO,HNA e MID é criada uma mensagem MAE do tipo AUTH;

– Todas as mensagens MAE estão no final do pacote, logo após as mensagens OLSR;

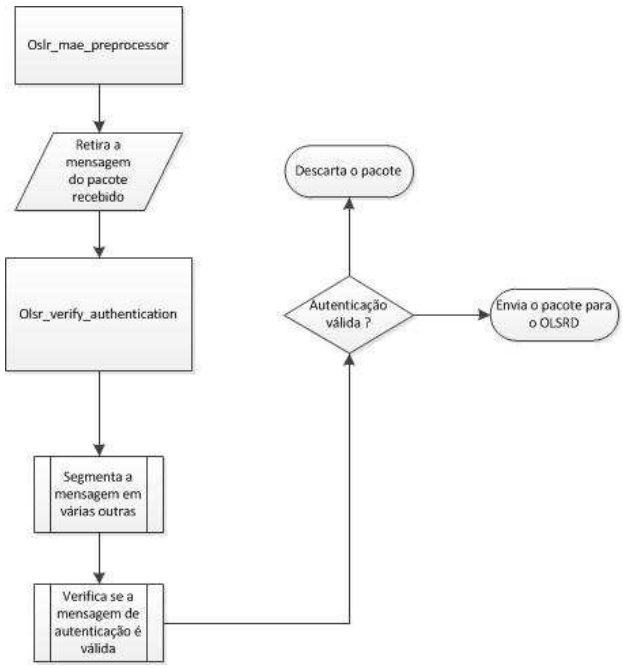

Figura 4.11: Diagrama de funções de entrada.

– Apenas uma mensagem do tipo CERT é adicionada por pacote e esta estará sempre no final;

– As mensagens do tipo HASH não foram implementadas, o cabeçalho está definido em mae.h.

A estrutura do pacote ficará a seguinte:

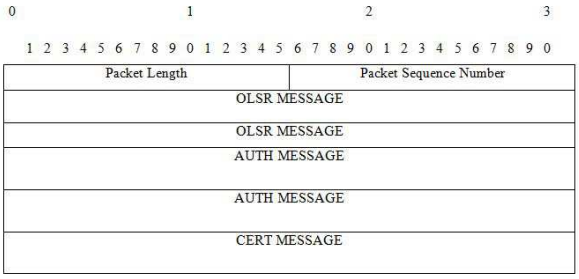

Figura 4.12: Formato de saída de pacote.

Exemplo com duas mensagens OLSR e dois AUTH abaixo mostrando a estrutura do pacote capturada utilizando um analisador de tráfego.

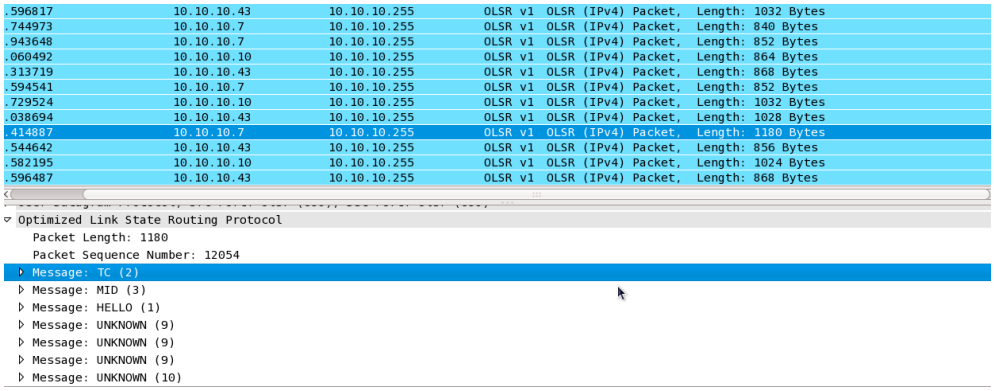

Figura 4.13: Pacote com três mensagens OLSR.

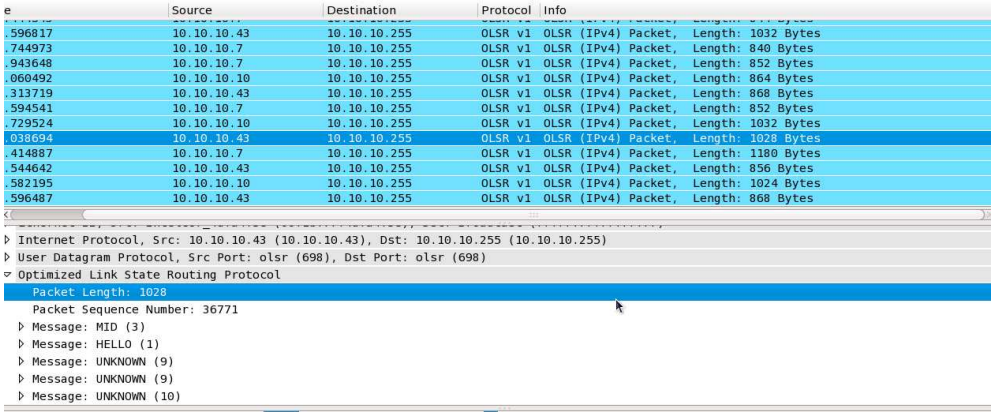

Figura 4.14: Pacote com duas mensagens OLSR.

5 CONCLUSÃO

Esse trabalho teve como proposta a implementação da Extensão de Autenticação para MANET para o Unik OLSR Daemon, seguindo o formato de mensagem da RFC 5444. A escolha do OLSRD foi motivada pelo fato de ser uma implementação do protocolo OLSR bastante difundida na comunidade e que permite extensões da sua funcionalidade por meio de plugins. Seu projeto modular é versátil o suficiente para que novas aplicações sejam introduzidas no protocolo, como um modelo de segurança para o protocolo, que é o escopo do trabalho. Muito dos protocolos de roteamento, mesmo os mais recentes, não prevêem uma maneira de transmitir suas mensagens de maneira confiável. Para sanar essas deficiências diversas propostas foram elaboradas, a maioria para protocolos específicos. Em há uma proposta de extensão de autenticação para MANET. Essa proposta prevê que as mensagens MANET devem ter objetos de autenticação que garantirão a autenticidade e integridade dessas mensagens, principalmente aquelas relevantes, como de protocolo de roteamento. A fim de ser uma implementação genérica para MANETs e não especifica para o protocolo OLSR, foi definido que as mensagens da MAE seguiriam o padrão da RFC 5444 que define um formato genérico para mensagens em MANETs. Nesse sentido, a implementação poderá ser estendida para outros protocolos e tipos de mensagem que necessitem de autenticação. O trabalho foi bem sucedido ao transformar a MAE em um plugin do OLSRD. Foi possível fazer a assinatura digital das mensagens mais relevantes do protocolo OLSR (MID, TC, HNA e HELLO), garantindo a autenticidade das mesmas. O protocolo OLSR fornece um mecanismo bastante eficiente para transmissão de mensagens em redes Ad Hoc. Utilizando a idéia do trabalho é possível estender a utilização da MAE para outros tipos de mensagem que utilizem o OLSR como meio de propagação.

5.1 TRABALHOS FUTUROS

Em trabalhos futuros pode-se aprimorar o plugin MAE com uma função que implementa a cadeia de hash para autenticar campos da mensagem mutáveis como TTL e Hop Count. Em trabalhos futuros, poderá ter campos que permita que o usuário escolha quais mensagens possam ser autenticadas utilizando a MAE. Em trabalhos futuros poderá ser implementado um serviço de certificação, permitindo um maior dinamismo na troca de certificados. Além de um serviço de certificação, a inclusão de uma função que permita a autenticação eficiente de campos mutáveis. Está previsto em [3], porém não foi implementado

nesse trabalho. Está previsto em [3] um serviço de autoconfiguração onde existe bastante pesquisa em fase inicial. Está previsto em [3] um sistema de detecção de intrusão que pode integrado com a MAE.

REFERÊNCIAS

[1] KURKOWSKI, T. C. S.; NAVIDI, W. *Two Standards for Rigorous MANET Routing Protocol Evaluation*.

[2] CORSON, S.; MACKER, J. *Mobile Ad hoc Networking (MANET):Routing Protocol Performance Issues and Evaluation Considerations*. IETF, jan. 1999. RFC 2501 (Informational). (Request for Comments, 2501). Disponível em: <http://www.ietf.org/rfc/rfc2501.txt>.

[3] PUTTINI, R. S. *Um Modelo de Segurança para Redes Móveis Ad Hoc.* (Tese de Doutorado).

[4] CLAUSEN, T.; JACQUET, P. *Optimized Link State Routing Protocol (OLSR)*. [S.l.], October 2003. Disponível em: <http://rfc.net/rfc3626.txt>.

[5] TONNESEN, A. *Implementing and extending the Optimized Link State Routing Protocol.*

[6] CLAUSEN, T. et al. *Generalized Mobile Ad Hoc Network (MANET)Packet/Message Format*. IETF, fev. 2009. RFC 5444 (Proposed Standard). (Request for Comments, 5444). Disponível em: <http://www.ietf.org/rfc/rfc5444.txt>.

[7] PACHECO, V. M. *Proposta e Implementação de uma MIB para o Protocolo OLSR.* (Dissertação de Mestrado).

[8] POSTEL, J. *Internet Protocol*. IETF, set. 1981. RFC 791 (Standard). (Request for Comments, 791). Updated by RFC 1349. Disponível em: <http://www.ietf.org/rfc/rfc791.txt>.

[9] DEERING, S.; HINDEN, R. *Internet Protocol, Version 6 (IPv6) Specification*. IETF, dez. 1998. RFC 2460 (Draft Standard). (Request for Comments, 2460). Updated by RFCs 5095, 5722. Disponível em: <http://www.ietf.org/rfc/rfc2460.txt>.

[10] POSTEL, J. *User Datagram Protocol*. IETF, ago. 1980. RFC 768 (Standard). (Request for Comments, 768). Disponível em: <http://www.ietf.org/rfc/rfc768.txt>.

[11] HAFSLUND ANDREAS TONNESEN, R. B. R. J. A. A.; KURE, O. *Secure Extension to the OLSR protocol*.