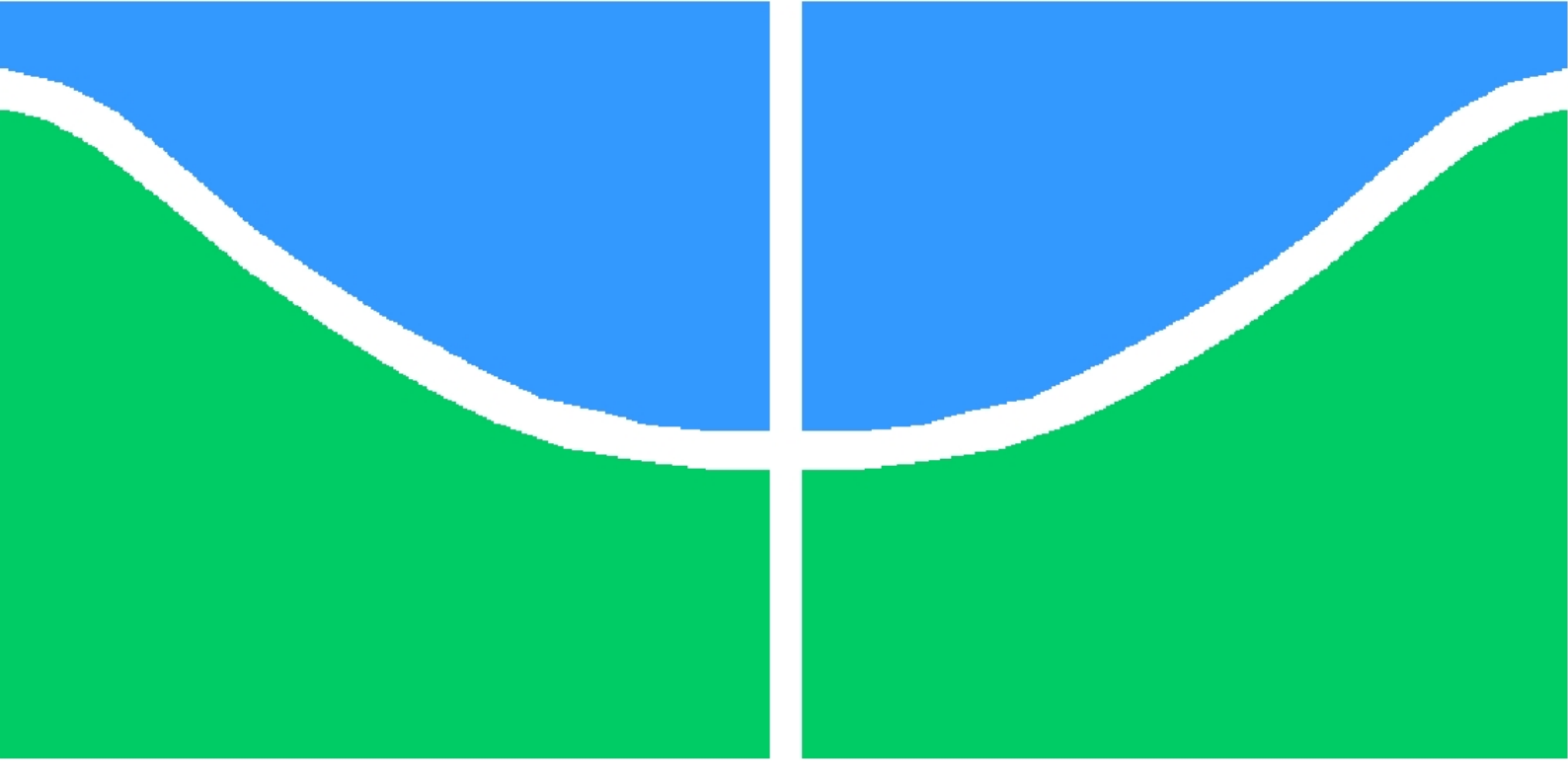

TRABALHO DE GRADUAÇÃO

Implementação de Serviço de Certificação Distribuída em Manet

Diego Barbosa Marques

Marcus Tôrres Silva

Brasília, setembro de 2010

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASILIA
Faculdade de Tecnologia

TRABALHO DE GRADUAÇÃO

Implementação de Serviço de Certificação Distribuída em Manet

Diego Barbosa Marques

Marcus Tôrres Silva

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 do Nascimento, ENE/UnB (Membro Interno) _____

Prof. Dr. Flávio Elias Gomes de Deus, ENE/UnB (Membro Interno) _____

FICHA CATALOGRÁFICA

Marques, D. B., Silva, M. T.
Implementação de Serviço de Certificação
Distribuída em Manet [Distrito Federal] 2010.
v, 26p. (ENE/FT/UnB, Engenheiro de Redes de Comunicação, 2010)
Monografia de Graduação - Universidade de Brasília. Faculdade de Tecnologia.
Departamento de Engenharia Elétrica.

1. Introdução
2. Redes MANET
3. Serviço de Certificação Digital Ditribuído (L-CERT)
4. Implementação
5.Conclusões

I. ENE/FT/UnB
II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA

Marques, D. B. e Silva, M. T. (2010). Implementação de Serviço de Certificação Distribuída em Manet Monografia de Graduação, Publicação ENE 01/2010, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 26p.

CESSÃO DE DIREITOS

NOMES DOS AUTORES: Diego Barbosa Marques e Marcus Tôrres Silva.

TÍTULO: Implementação de Serviço de Certificação Distribuída em Manet

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

Dedicatórias

Dedico à Deus, à familia e ao tio Hélio (in memoriam)

Diego Barbosa Marques

Dedico à Deus, à familia e aos amigos..

Marcus Tôrres Silva

Agradecimentos

Agradeço primeiramente à Deus, pois sem Ele nada disso seria possível. Agradeço ao nosso Prof. Dr. Ricardo Staciarini Puttini pela orientação neste trabalho, aos meus pais Adriles e Leuza, à minha irmã Lorena, e a todos que me ajudaram a completar mais esta etapa da vida.

Diego Barbosa Marques

Agradeço ao nosso orientador Prof. Dr. Ricardo Staciarini Puttini pela orientação neste trabalho. Aos meus pais: Antenor e Maria Helena. Aos meus irmãos: Fernanda e Junior. Ao meu sobrinho João Pedro e a todos os meus familiares. A Débora e sua família pelo constante apoio e incentivo. Aos meus amigos Bruno, Igor, João Paulo, João Rafael, Rodrigo, Thiago, Tancredo e a todos que ajudaram para que se concretizasse este sonho.

Marcus Tôrres Silva

RESUMO

Este trabalho tem como objetivo a implementação de um serviço de certificação digital em Manet. A implementação foi feita na forma de um plugin do OLSRD *(http://www.olsr.org)* cujos pacotes foram formatados segundo a especificação encontrada na RFC 5444 [1] e valendo-se dos algoritmos implementados em [2].

Abstract

This paper aims to implement a digital certification service in Manet. The implementation was donw in the form of a OLSRD plugin *(http://www.olsr.org)*, which packets are formatted according to the specification found in RFC 5444 [1] and drawing on the algorithms implemented in [2].

SUMÁRIO

LISTA DE FIGURAS

LISTA DE ACRÔNIMOS

MANET	*Mobile Ad Hoc Network*
BSS	*Basic Service Set*
AP	*Acess Point*
IP	*Internet Protocol*
AC	*Autoridade Certificadora*
OLSR	*Optimized Link State Routing Protocol*
OLSRD	*Optimized Link State Routing Daemon*
RFC	*Request for Comments*
WLAN	*Wireless Local Area Network*
WiMAX	*Worldwide Interoperability for Microwave Access*
WIP	*WiFi Phones*
PEM	*Privacy Enhanced Mail*

1 INTRODUÇÃO

As redes móveis *ad hoc*, também conhecidas como Manet, são redes sem fio que dispensam o uso de um ponto de acesso comum aos computadores conectados a ela. Desde modo, todos os dispositivos da rede podem funcionar como um roteador, encaminhando comunitariamente informações que vêm de dispositivos vizinhos, ou seja, os nodos móveis trocam informação sem auxílio de uma infraestrutura de rede pré-definida. Devido ao fato de não existir uma entidade central e pela dinâmica da topologia, o roteamento requer algoritmos distribuídos e adaptativos. Assim, os nodos dependem um dos outros para proverem os serviços de roteamento na rede.

Um aspecto importante das redes Manet é a segurança, que consiste em um desafio, devido a vários fatores como o fato dos nodos móveis utilizarem baterias (energia portátil) que acabam com o tempo e a comunicação ocorrer por meio de ondas de rádio, bem como o fato da topologia ser dinâmica e poder variar constantemente. Estas redes estão também sujeitas a vários tipos de ataques, que podem vir de várias direções e alvejar qualquer nó da rede, bastando que o nó malicioso esteja no alcance de transmissão do nó atacado.

Neste contexto, a segurança pode ser feita através da certificação digital, utilizada para identificação dos nodos confiáveis da rede e para proteger a troca de mensagens empregando esquemas de criptografia.

Este trabalho tem como objetivo a implementação de um serviço de certificação digital distribuída para Manet através de uma autoridade certificadora distribuída (AC). A distribuição das funcionalidades desta AC é baseada no protocolo criptográfico RSA [3] e realizada através do compartilhamento de sua chave privada entre todos os nodos participantes da rede utilizando a técnica de criptografia de limiar [4].

O serviço de certificação permite que, quando uma entidade solicita o serviço de certificação, um subgrupo (coalizão) de *k* portadores de chaves secretas é formado e cada nodo v_i fornece um certificado parcial assinado para a entidade que solicitou o serviço. Esta, por sua vez, combina os *k* certificados, obtendo o certificado completo, conforme visto em [5, 6, 2].

A implementação foi baseada nos formatos de mensagens especificados pela RFC 5444 [1] que descreve um formato de pacote projetado para transportar múltiplas mensagens dos protocolos de roteamento para troca de informações entre roteadores Manet.

O trabalho foi realizado em virtude da necessidade de prover segurança em Manet, por conseguinte

esta segurança precisa ser distribuída devido a alguns fatores já mencionados anteriormente. O trabalho foi realizado a partir da utilização da criptografia de limiar para prover um serviço de certificação digital, conforme proposto em [5, 6]. Para a implementação, nós utilizamos o *plugin* do OLSRD, que é um software que implementa o protocolo OLSR, de modo que quando o OLSR é executado inicia-se a propagação de rotas pelos nodos até a convergência ser atingida. Utilizamos também de algumas implementações de algoritmos realizadas em [2]. Nossa proposta é a implementação de um protocolo para prover esse serviço.

O restante deste trabalho está organizado da seguinte forma. No capítulo 2, a rede Manet é caracterizada, bem como os aspectos de segurança relativos a este tipo de rede. No capítulo 3, é apresentado a proposta do serviço de certificação e a estrutura dos formatos das mensagens utilizadas. Em seguida, no capítulo 4, são detalhadas as implementações utilizando o OLSRD. Por fim, no capítulo 5, são apresentadas as conclusões e as possibilidades de trabalhos futuros.

2 REDES MANET

Ultimamente notou-se um grande crescimento das redes sem fio, podendo ser citadas as redes locais sem fio, WIMAX, entre outras. Todas estas tecnologias são atraentes para os usuários que tem por objetivo uma rápida e simples instalação, sem contudo contar com os problemas que são encontrados nas redes cabeadas. As tecnologias de redes sem fio citadas acima não permitem que os dispositivos móveis comuniquem-se diretamente, o que já é possível através das redes móveis ad hoc (MANET). Os dois tipos de redes móveis mais utlizados são: as Redes Móveis Ad Hoc e as Redes BSS (*Basic Service Set*) também conhecidas por redes infraestruturadas.

2.1 REDES BSS

As redes BSS diferem-se das redes ad hoc pela necessidade da utilização de um ponto de acesso, por exemplo, um AP (Acess Point) ou uma estação base. Este ponto de acesso serve como ponte para conexão com a rede fixa, bem como para a comunicação entre dois dispositivos móveis que desejem se comunicar. Estas redes são formadas por um conjunto de estações sem-fio, controladas por um dispositivo coordenador denominado AP.

O funcionamento deste tipo de rede móvel é semelhante ao da telefonia celular, onde toda a comunicação deve, necessariamente, passar pela central, mesmo que os equipamentos móveis estejam a uma distância em que poderiam, eventualmente, comunicar-se diretamente.

As figuras 2.1 e 2.2 ilustram exemplos de rede BSS:

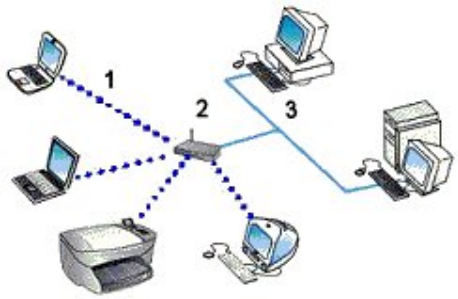

Figura 2.1: Redes BSS 1

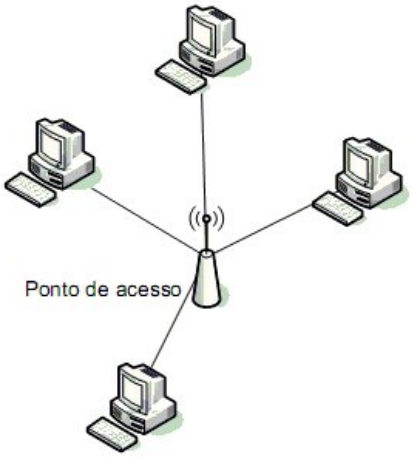

Figura 2.2: Redes BSS 2

2.2 REDES AD HOC

Em redes *Ad Hoc*, temos que todos os nodos possuem a habilidade de comunicarem-se, entre si, sem necessitar de um ponto de acesso para isto. Deste modo, os nodos são responsáveis por todas as funções da rede. Eles realizam a função de encaminhar as informações pela rede, como uma espécie de roteador, sendo também tarefa atribuída a eles a descoberta da melhor rota e a sua manutenção.

A utilização destas redes esta atrelada ao fato delas poderem ser facilmente instaladas, sendo que as mesmas permitem uma grande flexibilidade, pois não possuem uma estrutura física, fazendo com que seu uso seja bastante amplo, como em salas de aula, guerras, catástrofes, resgates, comércios, entre outros.

As figuras 2.3 e 2.4 ilustram exemplos de Redes Manet (Mobile Ad Hoc Network):

Figura 2.3: Redes Manet 1

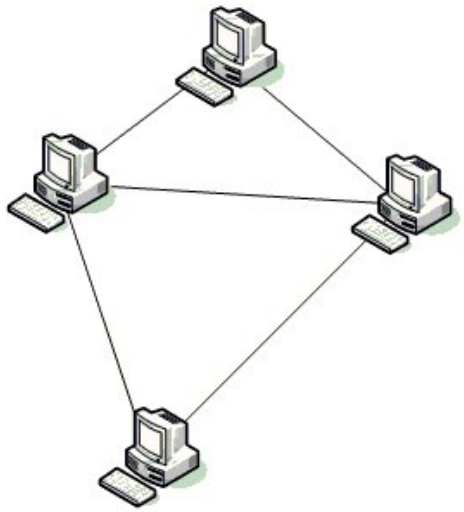

Figura 2.4: Redes Manet 2

2.3 SEGURANÇA EM MANET

Geralmente são considerados os seguintes atributos de segurança para redes de comunicação: disponibilidade, confidencialidade, integridade, autenticidade, e não-repúdio [7]. A disponibilidade é a característica conferida a rede de que mesmo sob algum tipo de ataque a rede continua em funcionamento. A confidencialidade assegura que certo tipo de informação não seja descoberta por entidades não autorizadas. A integridade deve garantir que uma mensagem enviada a outro nodo da rede não seja corrompida a não ser por falha da interface, mas nunca por comportamento de um nodo malicioso. A autenticidade deve capacitar os nodos de confirmar a identidade dos outros nodos participantes da rede de modo a evitar tentativas de alguns ataques, como a personificação. O não-repúdio confere ao sistema a capacidade de sempre identificar a origem de uma mensagem, o que acaba por ser de grande relevância do ponto de vista da necessidade de detectar nodos comprometidos.

2.3.1 Tipos de Ataques

Os ataques em redes ocorrem tanto nas redes infraestruturadas quanto nas Manet,uma vez que um adversário pode escutar promiscuamente o canal, comunicar-se diretamente com os outros nodos deste que esteja ao seu alcance, não colaborar com os nodos próximos, mesmo tendo "concordado"em fazê-lo, entre outros.

Posto isso, podemos ver que existem alguns tipos de ataques e os mesmos podem ser divididos em

ataques passivos e ativos. Os ataques passivos são aqueles em que um determinado nodo acaba por ignorar as funções que ele deveria realizar [8], agindo como se ele não estivesse ali. Já nos ataques ativos, temos um nodo malicioso que pode modificar, replicar ou colocar informações errôneas sobre a operação da rede, e ainda comporta-se de forma maliciosa e não cooperativa, afetando o sistema. A seguir, temos alguns tipos de ataques conhecidos:

- Ataques de Modificação: Nesse tipo de ataque, o nodo malicioso modifica a mensagem que foi recebida por ele, fazendo com que o mesmo possa acabar por receber todas as mensagens de trafego na rede ou até mesmo gerar informações errôneas, por exemplo com rotas falsas.

- Ataques de Fabricação: O ataque de fabricação é aquele em que um nodo fabrica mensagens de diversos tipos, por exemplo, fabricando mensagens com rotas falsas atraindo para si o tráfego.

- Ataques de Interceptação: O ataque de Interceptação tem como objetivo capturar o que esta sendo transmitido sem que o sistema perceba, ou seja, ataca-se a privacidade das informações.

- Ataques de Personificação - *Spoofing*: Diferente dos outros ataques, neste o nodo malicioso se faz passar por um nodo participante da rede, podendo obter variadas informações do trafego, mudar a rota de destino de um determinado pacote, entre outros.

Esses são alguns dos tipos de ataques que podem ocorrer nas redes, existindo ainda diversos outros ataques como o Ataque de Interrupção.

Alguns desses ataques explicitados acima, como o ataque de fabricação, de interceptação e o de personificação podem ser prevenidos com o uso da criptografia de limiar utilizada para prover um serviço de certificação digital distribuída, conforme será visto no capítulo 3.

2.3.2 Criptografia

A criptografia consiste de um conjunto de conceitos e técnicas que visa codificar uma informação de forma que somente o emissor e o receptor possam acessá-la, evitando que um intruso consiga interpretá-la. Para isso, uma série de técnicas são usadas e muitas outras surgem com o passar do tempo. Ela é útil em virtude de que caso alguém consiga interceptar a informação transmitida, este não irá conseguir interpretar a informação, pois ela não esta num formato que seja entendido sem o auxilio de ferramentas sofisticadas de quebra de criptografia.

A seguir iremos explicar alguns algoritmos de criptografia utilizados:

- Criptografia RSA: Este método de criptografia envolve um par de chaves para cada entidade, uma pública (KU) que é de conhecimento de todos e uma privada (KI) que deve ser mantida em sigilo. Toda mensagem cifrada usando uma chave pública só pode ser decifrada usando a respectiva chave privada.

- Assinatura Digital: A assinatura digital é baseada na criação de um código utilizando a chave privada, fazendo com que o nodo que receber uma mensagem contendo este código possa verificar se o remetente é mesmo quem diz ser e identificar qualquer mensagem que possa ter sido modificada. O seu funcionamento ocorre da seguinte maneira: Quando um nodo A quiser enviar uma mensagem para um nodo B, ele codificará o *hash* com sua chave privada, gerando assim uma assinatura digital que será adicionada à mensagem. Ao receber a mensagem o nodo B utilizará a chave pública de A para decodificar a mensagem. É importante ressaltar que o fato de assinar uma mensagem não significa gerar uma mensagem sigilosa. Na analogia apresentada temos que se o nodo A quiser assinar a mensagem e ter certeza de que apenas o nodo B teria acesso a seu conteúdo, seria preciso codificá-la com a chave pública de B, depois de assiná-la.

3 SERVIÇO DE CERTIFICAÇÃO DIGITAL DISTRIBUÍDO (L-CERT)

Para a implementação deste trabalho, utilizamos a especificação de formatos para pacotes que são usados por protocolos de roteamento para redes MANET, conforme a RFC 5444 [1] que será explicada adiante. Para esta implementação são considerados alguns requisitos genéricos de redes Manet, no qual é definido que os serviços de segurança operam segundo um modelo de serviços auto-organizados, distribuídos e cooperativos entre si.

Com a criptografia de limiar, podemos dividir uma chave privada em segredos parciais e recuperar a chave original através de *K* de N nodos. Usando dessa propriedade, foi proposta um serviço de certificação digital distribuída [5, 6] em que a chave da AC é dividida em N segredos parciais. A partir do momento que um nodo consegue formar uma coalizão de *K* nodos é possível calcular a assinatura de um certificado assinado pela AC, utilizando as assinaturas recebidas desses *K* nodos. Estas assinaturas são então computadas com o segredo parcial aplicado ao *hash* do certificado do nodo requerente. O trabalho realizado em [2], permite a utilização dos algoritmos necessários para calcular os segredos parciais e para calcular o certificado final assinado, com base nas assinaturas recebidas dos *K* nodos da coalizão, só que esse trabalho se utiliza de arquivos para isso. O *plugin* cria um protocolo para que isso seja possível, utilizando a matemática desenvolvida em [2] para fazer a emissão de certificados digitais. O capítulo seguinte explicará cada um desses conceitos utilizados.

Por praticidade, foi considerado que os nodos da rede já estão iniciados e colaborando mutuamente. Esta inicialização do sistema pode ser feita, por exemplo, por um negociador (*dealer*) que é uma entidade centralizada que existe somente no momento da inicialização, como demonstrado em [6]. O negociador gera o par de chaves RSA (KI_i, KU_i) e um polinômio randômico, utilizando a biblioteca OpenSSL.

Cada um dos N nodos iniciais da rede recebe uma chave parcial $KI_{AC,i}$ e, com isso a rede já pode funcionar. Após a rede ser iniciada, nem o nodo negociador e nem o polinômio são necessários mais para o funcionamento do sistema e são descartados. A partir deste momento, os nodos já iniciados são responsáveis pela emissão de certificados e de partes de KI_{AC} para nodos novos.

3.1 SERVIÇOS DE CERTIFICAÇÃO

Os Serviços de Certificação fornecem serviços personalizáveis para emissão e gerenciamento de certificados usados em sistemas de segurança que empregam tecnologias de chave pública. Uma autoridade certificadora (AC) pode ser usada para receber solicitações de certificados, verificar as informações na solicitação e a identidade do solicitante, bem como emitir cerificados. No caso em particular da Manet, a AC está presente apenas no momento da inicializaçãoo, conforme visto anteriormente.

Neste trabalho, este serviço fica disponível para todos os nodos da rede desde que os mesmos possam localizar uma coalizão com um número mínimo de nodos (K), de acordo com a criptografia de limiar, sendo a coalizão e esta criptografia explicadas na seção 3.2.

Como dito, o sistema já foi iniciado e o segredo de certificação (i.e. a chave privada da autoridade de certificação) já foi compartilhado entre os nodos participantes.

Com o sistema já iniciado e cada nodo possuindo uma parte da chave privada da autoridade certificadora pode-se, então, emitir certificados parciais para os nodos que desejam participar da rede, no qual a partir de (K) certificados parciais recebidos, o mesmo pode combiná-los para obter o certificado completo.

A provisão deste serviço é feita através de serviços locais de certificação (L-CERT), os mesmos são executados colaborativamente em todos os nodos da Manet. Para manter-se a robustez da soluçãoo, é requerido que um número mínimo de nodos (limiar) concordem (noção de política de segurança) e cooperem para que um novo nodo seja admitido na rede (serviço de certificação).

3.2 CRIPTOGRAFIA DE LIMIAR

A Criptografia de Limiar é um ramo da criptografia que estuda como compartilhar o segredo ou função criptográfica entre um grupo de computadores, conforme pode ser visto em [4]. Neste modelo, o segredo de certificação D pode ser quebrado em diversas partes N, na qual cada parte é atribuída a um determinado conjunto de nodos participantes, de forma que D pode ser reconstruído a partir de K partes, mas o conhecimento de K-1 partes não revela nenhuma informação de D.

Este número K é um valor de muita importância para o sistema e representa um compromisso entre o nível de segurança requerido e a escalabilidade do mesmo.Tendo em vista que quanto maior for o valor de K, mais robusto será o sistema com relação à existência de nodos comprometidos, pois são requeridos pelo

menos K nodos comprometidos e colaborando entre si para que a segurança do serviço de certificação seja quebrada. Aqui pode-se notar que a disponibilidade e a escabilidade do sistema pode ficar comprometido dependendo de quão grande for este valor. Em outra vertente, temos que um valor pequeno de K acarreta em um menor *overhead* de comunicação necessário para a provisão dos serviços. No caso limite, onde K=1, temos que um nodo único pode quebrar o sistema, ficando ele potencialmente inseguro.

Conforme [6], temos que neste modelo a constante K indica o número de nodos que devem confiar em um outro nodo para que este possa ser admitido na rede e N é o número (não fixo) de nodos que possuem uma parte da chave privada de certificação. Cada nodo que possua uma parte da chave privada da AC pode, então, emitir certificados parciais para os nodos nos quais ele confia. Estes nodos, por sua vez, recuperam quaisquer K certificados parciais emitidos em seu favor e podem combiná-los para obter o certificado completo. Para que um determinado nodo possa obter este certificado ele precisa formar uma coalizão de pelo menos K nodos que possuam segredos parciais. A distribuição de partes da chave privada de certificação e a revogação de certificados são igualmente realizadas colaborativamente.

3.3 RFC 5444

Este documento especifica o formato que um pacote deve ter para ser usado pelos protocolos de roteamento em redes MANET.

Estes pacotes são projetados para transportar múltiplas mensagens e são compostos por um cabeçaalho que pode conter uma ou mais mensagens. A mensagem consiste em um cabeçalho, para controle da disseminação da mesma, e um corpo que contém os atributos associados a ela.

Os pacotes <packet> são definidos por:

- <packet-header>

- <message>

O <packet-header>, que pode ser visto na figura 3.1, é definido por:

- <version>

- <pkt-flags>

- <pkt-seq-num>

- <tlv-block>

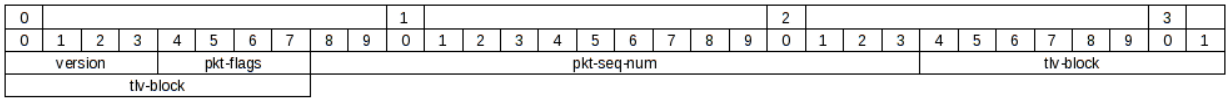

Figura 3.1: Formato do cabeçalho do pacote

Onde:

<version>: Campo de 4 bits que especifica a versão na qual o pacote e as constantes são construídas.

<pkt-flags>: Campo de 4 bits que funciona da seguinte maneira:

- Bit 0: Se tiver seu valor como (’0’), então <packet-seq-num> não é adicionado no <pkt-header>, e se for (’1’) então <pkt-seq-num> é adicionado no <packet-header>.

- Bit 1: Se tiver seu valor como (’0’), então <tlv-block> não é adicionado no <packet-header>, e se for (’1’) então <tlv-block> é adicionado no <packet-header>.

- Bit 2 e 3: são reservados, e devem ter seus valores em (’0’) na transmissão e devem ser ignorados na recepção.

<pkt-seq-num>: Campo de 16 bits, que indica o número da sequência do pacote.

<tlv-block>: que é composto por dois campos, <tlvs-length> <tlv>, no qual não entraremos em detalhe nestes, pois eles não são utilizados neste trabalho.

Para as mensagens <message>, temos que o cabeçalho <msg-header> (figura 3.2) é definido por:

- <msg-type>

- <msg-flags>

- <msg-addr-length>

- <msg-size>

- <msg-orig-addr>

- <msg-hop-limit>

- <msg-hop-count>

- <msg-seq-num>

Onde:

<msg-type>: Campo de 8 bits que especifica o tipo da mensagem.

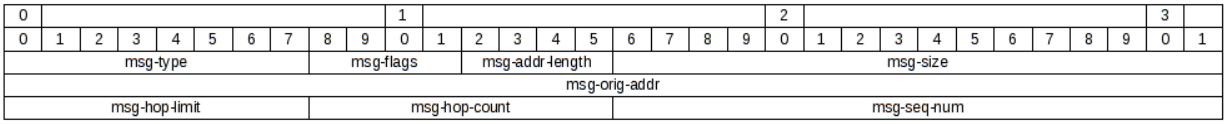

Figura 3.2: Formato do cabeçalho da mensagem

<msg-flags>: Campo de 4 bits que funciona da seguinte maneira:

- Bit 0: Se tiver seu valor como (’0’), então <msg-orig-addr> não é adicionado na <msg-header>, e se for (’1’) então <msg-orig-addr> é adicionado na <msg-header>.

- Bit 1: Se tiver seu valor como (’0’), então <msg-hop-limit> não é adicionado na <msg-header>, e se for (’1’) então < msg-hop-limit > é adicionado na <msg-header>.

- Bit 2: Se tiver seu valor como (’0’), então <msg-hop-count> não é adicionado na <msg-header>, e se for (’1’) então < msg-hop-count > é adicionado na <msg-header>.

- Bit 3: Se tiver seu valor como (’0’), então <msg-seq-num> não é adicionado na <msg-header>, e se for (’1’) então < msg-seq-num > é adicionado na <msg-header>.

<msg-addr-length>: Campo de 4 bits que codifica o cumprimento de todos os endereços na mensagem (<msg-orig-addr>), bem como cada endereço incluído no bloco de endereços, como a seguir:

<msg-addr-length> = o comprimento do endereço em octeto - 1, deste modo ele será 3 para endereços IPV4 e 15 para endereços IPV6.

<msg-size>: Campo de 16 bits que especifica a quantidade de octetos que compõe a mensagem (<message>), incluindo o cabeçalho (<msg-header>).

<msg-orig-addr>: É um identificador com comprimento igual ao comprimento do endereço do roteador que originou a mensagem.

<msg-hop-limit>: Campo de 8 bits que contem o número máximo de saltos (*hop‘s*) que uma mensagem pode ter até chegar a seu destino, antes de ser descartada.

<msg-hop-count>: Campo de 8 bits que pode conter o número de saltos (*hop‘s*) que a mensagem levou para ir da origem até o destino.

<msg-seq-num>: Campo de 16 bits, que pode conter um número de sequência gerado pelo originador da mensagem.

3.4 PROPOSTA DE PROTOCOLO L-CERT

O trabalho realizado consiste na proposta e implementação do modelo de emissão e renovação de certificados digitais, a partir de trocas de mensagens em conformidade com [1] e com as necessidades do caso em particular, sendo que as estruturas das mensagens serão vistas posteriormente.

A partir do momento que o sistema já esteja iniciado e funcionando, um nodo que não esteja certificado ou que esteja necessitando renovar o seu certificado pode solicitar um novo certificado para outros nodos que ofereçam o serviço de certificação, sendo necessário que ele forme uma coalizão de pelo o menos (*K*) nodos.

3.4.1 Protocolo L-CERT

Este serviço é realizado conforme proposto em [6], sendo feito em quatro etapas: (1) e (2) formação da coalizão, (3) requisição do serviço, e (4) coleta e processamento das respostas. Explicitando cada uma das etapas, temos:

1 - Quando um nodo (v_i) necessita receber um certificado, ele dissemina uma mensagem de requisiçãoo de coalizão (COALITION_REQ) contendo REF_N, onde REF_N é o número de referência da transação, que é gerado aleatoriamente e deve estar presente em todas as demais mensagens do processo.

2 - Qualquer nodo que possui uma parte de KI_{AC} deve responder à requisição enviando uma mensagem autenticada de notificação de coalizão (COALITION_ACK) ao requerente, contendo sua identidade v_j e REF_N.

3 - O nodo requerente coleta respostas (COALITION_ACK) até que seja possível formar uma coalizão de *K* nodos. O próprio requerente pode fazer parte da coalizão, caso ele possua uma parte de KI_{AC}. O conjunto de identidades dos nodos da coalizão é dado por $\beta = \{v_i/v_j \in \text{coalisão}\}$. Em seguida, o nodo dissemina uma mensagem de requisição de certificado (CERT_REQ), contendo a requisição do certificado, $< v_i, KU_i, T_{sign}, T_{expire} >$, onde v_i é o identificador do nodo,KU_i é sua chave pública, T_{sign} é um selo de tempo (*timestamp*) com a data e hora do início de validade do certificado e T_{expire} é um selo de tempo com a data e hora da expiração do certificado. No caso da emissão de um certificado para um nodo que não tenha um certificado válido, as informações de identidade do nodo requeridas pela polética de certificação são adicionadas à requisição.

4 - Ao receber CERT_REQ, os nodos que fazem parte da coalizão identificam sua própria identidade

na coalizão. A política de certificação apropriada é aplicada e cada nodo da coalizão decide se atende à requisição. Em caso afirmativo, o certificado parcial CERTi,j é calculado e enviado ao requerente, em uma mensagem PARTIAL_CERT. A figura 3.3 ilustra o procedimento descrito.

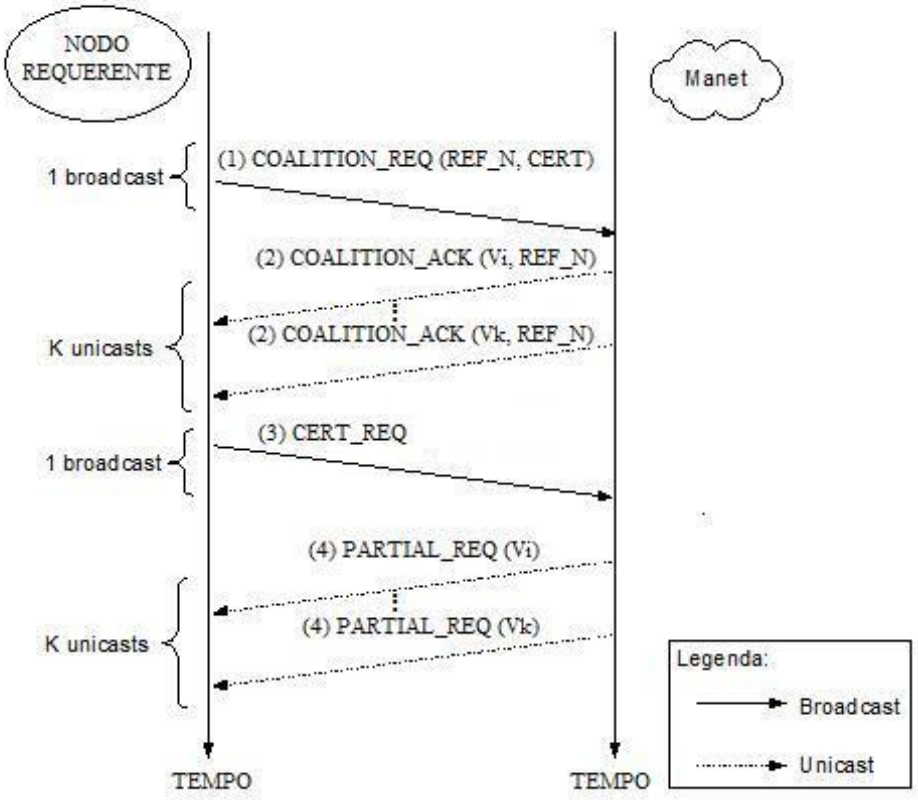

Figura 3.3: Protocolo L-CERT

O cálculo de $CERT_{ij}$ é feito da seguinte forma, de acordo a com [6]. Cada nodo v_j que resolve atender a uma requisição de v_i, calcula sua chave aditiva $KI_{j,\beta}$,conforme Eq.3.1:

$$KI_{j,\beta} = KI_{CA,j} L_{j,\beta}(0) = KI_{CA,j} \prod_{je\beta, r \neq j} \frac{V_r}{V_r - V_j} \mathrm{mod} n_{AC} \tag{3.1}$$

em que: $L_{j,\beta}(V_i)$ são os coeficientes de Lagrange para interpolação do polinômio gerador f(x), dados por Eq.3.2:

$$L_{j,\beta}(V_i) = \prod_{je\beta, r \neq j} \frac{V_i - V_r}{V_j - Vr} \tag{3.2}$$

$KI_{j,\beta}$ é chamada chave aditiva, pois pela interpolação de Lagrange tem-se:

$$\sum_{re\beta} KI_{r,\beta} = \sum_{re\beta} KI_{AC,r} L_{r,\beta}(0) = d_{AC} \text{mod} n_{AC} = t.n_{AC} + d_{AC} \tag{3.3}$$

onde: $0 \leq t < k$.

Em seguida, v_j calcula o certificado parcial de $v_i(CERT_{i,\beta})$, assinando o *hash* do novo certificado $(cert_i)$ com a chave aditiva $KI_{j,\beta}$ Eq.3.4:

$$CERT_{i,j} = (cert)^{KI_{j,\beta}} \text{mod} n_{AC} \tag{3.4}$$

Finalmente, após receber os *K* certificados parciais, v_i combina-os para gerar um candidato para a assinatura do certificado $(CERT_i')$, conforme a Eq.3.5:

$$CERT_i' = \prod_{re\beta} CERT_{i,r} \text{mod} n_{AC} = (cert_i)^{\sum_{re\beta} KI_{i,r}} \text{mod} n_{AC} = (cert)^{t.n_{AC}+d_{AC}} \text{mod} n_{AC} \tag{3.5}$$

Observando que $CERT_i = (cert)^{d_{AC}} \text{mod} n_{AC}$, verifica-se que este candidato $CERT_i'$ é diferente de $CERT_i$ por uma constante. Esta polarização pode ser removida pelo algoritmo da figura 3.4, visto em [6].

Algoritmo 1 – Cálculo de $CERT_i$

Entradas: $CERT_i'$ (candidato a assinatura do certificado) e $cert_i$ (*hash* do certificado a ser assinado).

Saída: $CERT_i$ (assinatura do certificado).

1: $Z := (cert_i)^{-e_{AC}} \bmod n_{AC}$
2: $r := 0, Y := CERT_i'$
3: *while* $j < K$ *do*
4: $\quad Y := Y \cdot Z \bmod n_{AC}; j := j + 1$
5: $\quad$ *if* $(cert_i = Y^{e_{AC}} \bmod n_{AC})$ *then*
6: $\quad\quad$ *break while*
7: $\quad$ *end if*
8: *end while*
9: *saída*: $Y = CERT_i$

Figura 3.4: Algoritmo de Remoção de Polarização

Se algum dos nodos da coalizão não enviar o (PARTIAL_REQ), ou um dos nodos enviar o certificado

parcial e ele não for recebido pelo nodo requisitante, ou ainda se ocorrer uma falha em algum dos nodos, os certificados parciais dos outros nodos tornam-se sem utilidade, sendo necessário reiniciar todo o processo.

3.4.2 Estrutura das Mensagens

As mensagens utilizadas no trabalho foram feitas em conformidade com [1] e as mesmas tem o aspecto mostrado na figura 3.5:

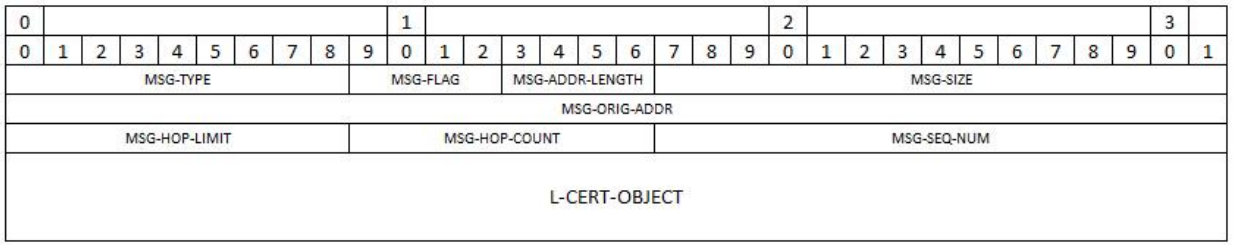

Figura 3.5: Estrutura da Mensagem utilizada no serviço de certificação

Podendo ser visto, que o valores dos bits de <msg-flags> foram selecionados em (’1’) para que <msg-orig-addr>, <msg-msg-hop-limit>, <msg-hop-count> e <msg-addr-lenght> fossem adicionados.

O valor do campo <msg-type> foi escolhido arbitrariamente como sendo o número 222.

Já o valor de <msg-addr-lenght> foi selecionado em 3 devido ao fato de estarmos utilizando IPV4, bem como pode ser visto que na parte do corpo da mensagem foi colocado o L-CERT-OBJECT, que contem os tipos de mensagens que vão ser trocadas entre o nodo requisitante e o requisitado, sendo importante ressaltar que o cabeçalho da mensagem é o mesmo em todas as trocas de mensagens, apenas mudando o L-CERT-OBJECT.

Posto isto, o formato de L-CERT-OBJECT, vai variar de acordo com o protocolo de comunicação explicado acima. Para a primeira troca de mensagens, que ocorre quando o nodo requisitante envia o COALISION_REQ, temos o formato mostrado na figura 3.6:

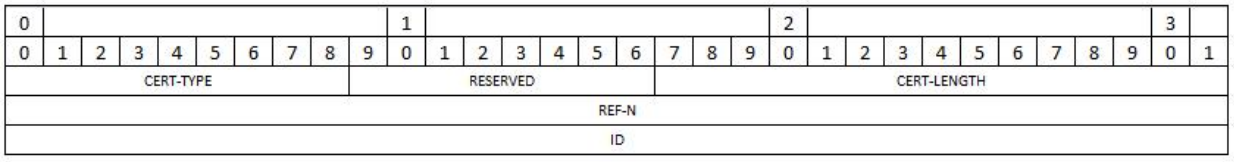

Figura 3.6: Estrutura da Mensagem COALISION_REQ e COALISION_ACK

Para a segunda troca de mensagens, que contempla a mensagem COALISION_ACK, temos o mesmo formato mostrada na figura 3.6, em virtude de que estas duas primeiras trocas de mensagens, ocorrem com

o intuito de que o nodo requisitante possa coletar os ID‘s dos nodos requisitados para conseguir formar a sua coalizão, conforme explicado anteriormente.

Na terceira troca de mensagens, o nodo requisitante envia a mensagem CERT_REQ, com o formato mostrado na figura 3.7.

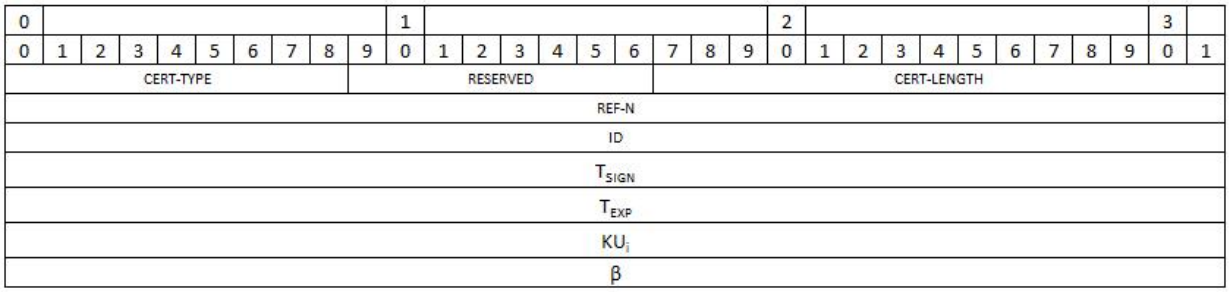

Figura 3.7: Estrutura da Mensagem CERT_REQ

E por fim os nodos que receberam estes, enviam o PARTIAL_REQ, figura 3.8.

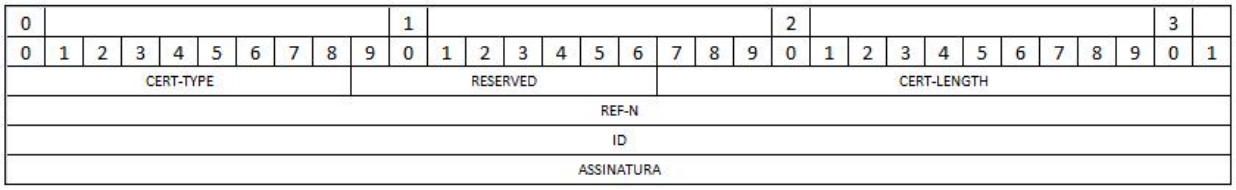

Figura 3.8: Estrutura da Mensagem PARTIAL_REQ

4 IMPLEMENTAÇÃO

4.1 OLSRD

Para atender as especificações do protocolo OLSR descrito na RFC 3626 [9], se fez necessária a implementação de um programa que pudesse atender a tais especificações. Neste contexto, surgiram algumas implementações como o NRL OLSR, OOLSR e o OLSRD. Dentre estas a que mais se destacou foi o OLSR Daemon (OLSRD). Devido a sua popularidade, arquitetura e possibilidade de novas extensões, foi utilizado o OLSRD neste trabalho. A seguir iremos descrever as suas principais funcionalidades bem como a interface *plugin* - OLSRD.

4.1.1 Visão Geral

O início de seu desenvolvimento se deu com o GNU/LINUX feito apenas para suportar este tipo de plataforma, mas atualmente o mesmo já esta disponível para a plataforma Windows, OS X, WIP (*WiFi Phones*), entre outros. O seu desenvolvimento seguiu alguns princípios, conforme explicado e visto em [10]:

- **Modularidade**: Para que um código possa ser utilizado por qualquer entidade e também para que ele possa ser entendido como um mecanismo geral se faz necessário que o mesmo seja o mais modular possível. O que significa que as entidades que utilizam as suas funções devem se registrar dinamicamente na função.

- **Estrutura de dados consistente**: Como visto na RFC 3626 [9] o protocolo OLSR é baseado em tabelas e devido a isto todos os dados contidos têm de ser consistentes e as tabelas devem possuir a mesma estrutura.

- **Transparência IP**: O Daemon deve ser tão transparente quanto possível, o que significa que, tanto quanto possível o código deve trabalhar em ambos os tipos de endereços: IPv4 e IPv6.

- **Código legível**: O código tem de ser de fácil leitura para um observador, o que nem sempre é fácil de conseguir, pois um programador tem uma visão muito subjetiva de seu próprio código.

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

Posto isso, a figura 4.1 ilustra a implementação OLSR daemon baseada em [11], sendo as suas respectivas entidades explicadas brevemente.

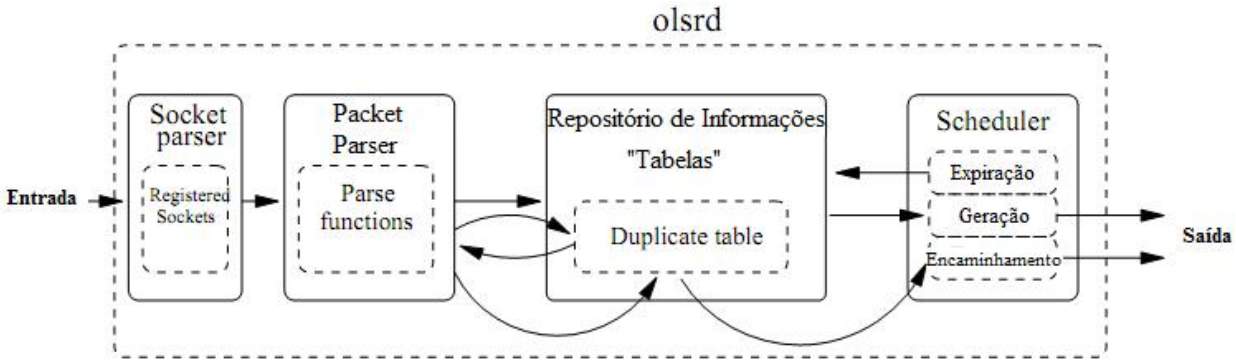

Figura 4.1: Estrura de Funciomento do OLSRD

- **Socket Parser**: Todo tráfego que chega ao OLSR passa primeiramente por esta entidade. O Socket Parser checa todo este tráfego utilizando um loop. Em seguida ele solicita a função associada ao socket que tem o dado que esta chegando.

- **Packet Parser**: Esta entidade tem basicamente três opções para tratar uma mensagem quando recebida por ela que são:

1 - Descarta o pacote. Isto é feito se o pacote é considerado inválido.

2 - Processa o pacote de acordo com as instruções dadas. Para fazer isso, o *parser* tem de ser capaz de tratar este tipo de mensagem.

3 - Encaminha o pacote de acordo com o algoritmo de encaminhamento padrão. Isto é feito se o pacote for válido, mas o *parser* não tem conhecimento deste tipo de mensagem.

- **Repositório de Informações**: Como dito anteriormente o OLSR é um protocolo baseado em tabelas. Isso significa que a informação é atualizada e removida dinamicamente com mudanças na topologia da rede entre outros fatores. As tabelas podem ser consideradas o coração dos protocolos de roteamento. É neste contexto que o OLSRD implementa os repositórios de informações. Nestas tabelas são armazenadas as informações e todos os cálculos de rotas são feitos com base nesses repositórios. A funcionalidade de transmissão em particular, baseia-se nas tabelas duplicadas (*Duplicate table*) que é um *cache* de todos os pacotes encaminhados e/ou processados. Sempre que esses dados são atualizados de forma que muda o entendimento da topologia da rede, as rotas são recalculadas.

-**Scheduler**: Esta entidade executa diversos eventos em diferentes intervalos. Se uma certa entidade deseja que alguma tarefa seja realizada em intervalos regulares, deve-se registrar uma função no *scheduler*

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. Em seguida, as funções registradas que devem ser executadas são chamadas e mudanças de topologia e vizinhança são processadas.

4.1.2 OLSRD Plugins

Como pôde ser visto o OLSRD permite que se faça extensões do funcionamento do OLSR por meio de *plugins*. Devido ao fato do código ser modular, conforme exposto anteriormente, os *plugins* podem ter acesso a quase todas as funções da implementação, o que os tornam de grande valia para diversos usos. Entre as vantagens que tornaram seu uso viável, destacam-se:

- Não há necessidade de alterar qualquer código no OLSR daemon para adicionar pacotes personalizados ou funcionalidades.

- *Plugins* podem ser escritos em qualquer linguagem que implemente biblioteca dinâmica.

- Os usuários estão livres para implementar *plugins* OLSRD e licenciá-los sob qualquer termo que eles desejem.

A figura 4.2 ilustra como pode ser descrita uma interface do *plugin*, baseada em [11].

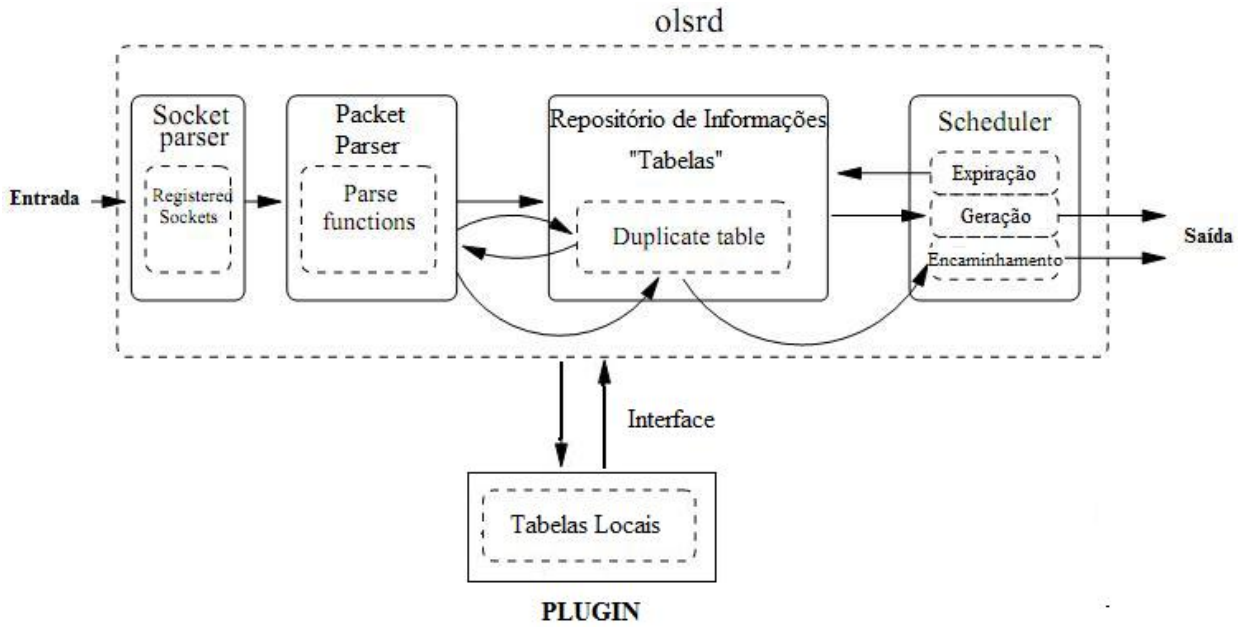

Figura 4.2: Esquema do Plugin

Devendo ser notado que a interface entre o *plugin* e o *daemon* permite que o *plugin* possa sempre saber o que esperar do *daemon* e o *daemon* do *plugin*. Deste modo sempre que ocorrer um determinado evento o

daemon chamará o *plugin* para buscar uma variável ou para executar uma determinada função.

4.2 IMPLEMENTAÇÃO PLUGIN PARA EMISSÃO DE CERTIFICADOS DIGITAIS

Para que fosse possível a implementação da emissão de certificados digitais para o OLSRD, utilizamos o modelo de plugin conforme explicado e visto em [10]. De forma geral, o plugin funciona da maneira explicada no capítulo 3 e foi implementado da seguinte maneira:

Ao executar o OLSRD, o plugin verifica se o nodo possui todos os arquivos necessários ao funcionamento. Esses arquivos são descritos na seção 4.3. Se um nodo possui um certificado válido, nenhum procedimento é tomado. Se além de um certificado válido o nodo possui um segredo parcial, ele é habilitado a ser participante de uma coalizão, podendo assim assinar requisições de certificados.

Após essa verificação inicial, o plugin registra a função que trata das mensagens do serviço de certicação no Packet Parser (figura 4.2). Dessa forma todas as mensagens do formato L-CERT serão encaminhadas ao plugin.

Caso o nodo não possua um certificado válido é iniciado o processo de requisição de certificado, onde é disseminada em broadcast a mensagem COALISION_REQ até que se obtenha K respostas COALISION_ACK, vindas de nodos distintos.

Ao obter essas respostas, é formada a coalizão. O nodo requisitante então envia aos nodos da coalizão a mensagem CERT_REQ. Diferente da proposta em [6], enviamos apenas os parâmetros mutáveis para a criação do certificado : $< v_i, T_{sign}, T_{expire}, KU_i >$. Os demais parâmetros devem previamente acordados e definidos na política de segurança da rede.

Cada nodo que receber a mensagem CERT_REQ verifica se faz parte da coalizão. Caso positivo, ele utiliza os parâmetros recebidos para criar o certificado do nodo requisitante e assina esse certificado com seu segredo parcial. Em seguida, o nodo responde a requisição enviando a assinatura do certificado criado através da mensagem PARTIAL_CERT. Novamente houve uma alteração em relação à proposta inicial [6], onde todo o certificado era enviado.

Por fim, o nodo requisitante calcula a assinatura do seu certificado utilizando as assinaturas recebidas e a chave pública da AC. O certificado final é gravado em um arquivo no formato PEM.

O critério para envio de respostas COALISION_ACK e PARTIAL_CERT deve ser definido na política de segurança da rede. Na implementação, todas as requisições são respondidas. Pode-se notar aqui uma

vulnerabilidade da rede pois um nodo malicioso pode disseminar infinitas requisições com o objetivo de sobrecarregar a rede e os nodos detentores de segredos parciais, gerando negação de serviço.

Os códigos relativos à implementação do trabalho foram publicados no SourceForge.net, estando assim disponíveis para teste e utilização da aplicação. A endereço do projeto é *http://sourceforge.net/projects/lcertmanet/*

4.3 UTILIZAÇÃO DO PLUGIN

Para a utilização do plugin desenvolvido, são necessários alguns requisitos. Primeiramente, todos os nodos precisam de um ID. Esse id deve ser armazenado no arquivo "id", e consiste em uma sequencia de 32 bits. Os nodos que possuírem segredo parcial devem necessariamente possuir esse arquivo. Para os nodos sem segredo parcial e sem id, o programa gera um numero de identificação aleatoriamente, e grava esse ID em arquivo. Também é necessário que todos os nodos possuam um par de chaves RSA, com o número de bits definidos pela política de segurança da rede.

Para iniciar a rede, utiliza-se o programa *dealer*. Ele é responsável gerar os segredos parciais necessários para inicializar o sistema. Ele deve receber como parâmetro o valor de k , o número de bits para geração da chave secreta e os caminhos IDs (número de identificação dos nodos) que vão receber um segredo parcial. Com a informação do número de bits o par de chaves é gerado. Com o valor de k e a chave privada o software pode gerar o polinômio secreto. Em seguida com polinômio gerado e os IDs é possível gerar os segredos parciais.

O software deve receber os parâmetros da seguinte forma:

-out <caminho> : Caminho para os arquivos de saída. Os arquivos de saída são:

E.key : chave publica.

N.key : valor de n.

<nome do id>.key : chave secreta parcial para o id. Os arquivos serão gravados com os valores em hexadecimal.

-k <valor> : Número mínimo de segredos parciais necessários para descobrir KI .

-nbits <valor> : Número bits para geração do par RSA.

-pk <caminho> : Caminho para o arquivo que contem a chave publica para a geração do par RSA. Não é um parâmetro obrigatório. Se for omitido o valor de pk será obtido randomicamente.

-ids <caminho> <caminho>... : Caminhos para os arquivos que contêm os ids dos nodos que receberão os segredos parciais.

Ex: **dealer -k 4 -nbits 1024 -ids ./id1 ./id2 ./id3 ./id4 ./id5 -out ./keys/**

Neste exemplo o *Dealer* vai gerar os seguintes arquivos: ./keys/E.key , ./keys/N.key , ./keys/id1.key , ./keys/id2.key , ./keys/id3.key , ./keys/id4.key , ./keys/id5.key

Para recuperar o segredo *KI* serão necessárias 4 das 5 chaves geradas.

Todos os arquivos necessários devem ser mantidos em uma mesma pasta, definida no arquivo de configuração do OLSRD. As chaves públicas dos nodos devem ser protegidas por senha e gravadas no formato PEM. Os certificados assinados também devem ser gravados no formato PEM e protegidos por senha para KI.

Como todos os processos descritos acima realizados, a rede está pronta para iniciar. Ao final do processo, deve ser criado um arquivo chamado *cert_signed.pem*, que contém o certificado digital assinado pela AC. Esse arquivo é gravado na pasta definida no arquivo de configuração.

5 CONCLUSÕES

Neste trabalho fez-se a proposta e implementação de um serviço capaz de prover a emissão e renovação de certificados digitais através de uma autoridade certificadora completamente distribuída e auto-organizada. A solução foi construída na forma de plugin para o OLSRD e possibilita que, dada uma rede iniciada e configurada seguindo as diretrizes apresentas, um novo nodo possa interagir com qualquer grupo arbitrário de nodos, detentores de K-de-N segredos parciais da rede, e assim obter um certificado digital assinado.

A concepção teve como base o uso de criptografia de limiar aplicada ao compartilhamento de chaves parciais, proposta inicialmente em [4], desenvolvida em [5, 6, 2], e que teve sua implementação iniciada em [2]. Além disso, arquitetou-se todas as mensagens utilizadas no protocolo em conformidade com a RFC 5444 [1], seguindo assim as tendências do *Workgroup* de Manet junto à IETF.

O trabalho acrescentou aos trabalhos citados anteriormente contribuições importantes, entre elas: a definição do protocolo de comunicação para a emissão e renovação de certificados digitais, a arquitetura de mensagens em conformidade com o formato generalizado de pacotes/mensagens para Manet [1], publicação do código do software desenvolvido em um repositório de acesso público.

A despeito de todo o desenvolvido, restaram melhorias a serem implementadas, passíveis de trabalhos futuros. Entre elas, destacamos:

1 - No momento em que o nodo requisitante recebe as mensagens PARTIAL_REQ dos requisitados é possível fazer a verificação destes certificados a partir do demonstrado na seção 5.4 do trabalho visto em [5], sendo também possível assinar as mensagem COALISION_ACK.

2 - Implementação da distribuição dos segredos parciais, conforme explicado e assumido no *caput* do capítulo 3 do presente trabalho e demonstrado na seção 6 do trabalho visto em [5].

3 - Pode-se notar que a coalizão só pode ser formada com os vizinhos de 1 salto. Caso ele não consiga formar esta coalizão, uma das soluções possíveis seria aumentar o número de saltos do COALISION_REQ, para que o mesmo consiga então formar uma coalizão com nodos que estejam a mais de 1 salto de distância dele.

Cabe ressaltar que estudos voltados para a definição de políticas de segurança também são essenciais para a eficácia de qualquer protocolo ou modelo de segurança proposto para redes ad hoc.

Em fim, este é apenas um ramo da criptografia, sendo visto que a demanda por estes serviços tendem a aumentar, alem de que atualmente existem poucas implementações feitas neste sentido.

REFERÊNCIAS

[1] T. Clausen, C. Dearlove, J. Dean and C. Adjih - *Generalized Mobile Ad Hoc Network (MANET) Packet/Message Format - IETF RFC 5444*, February 2009.

[2] F. Silveira e M. Hanashiro - *Serviços de Certificação para Redes Móveis Ad Hoc*, Monografia de projeto final, publicação UnB.LabRedes.PFG.02/2003, Departamento de Engenharia de Redes de Comunicação, Universidade de Brasília, Brasília - DF, 2003.

[3] . Rivest, A. Shamir and L. Adleman - *A method for obtaining digitial signatures and public key cryptosystems, Communications of the ACM*.Communications of the ACM, Feb 1978.

[4] A. SHAMIR - *How to Share a Secret*. Communications of the ACM, 22(11):612-613, 1979.

[5] H. Luo and S. Lu - *Ubiquitous and Robust Authentication Services for Ad Hoc Wireless Networks*. [S.l.: s.n.], UCLA Computer Science Technical Report 200030, Oct. 2000

[6] PUTTINI, R.S. - *Um modelo de Segurança para redes móveis Ad Hoc*. Tese (Doutorado) - Universidade de Brasília, 2004.

[7] L. Zhou and Z. J. Haas - *Securing ad hoc networks*. IEEE Network Magazine, November/December 1999.

[8] F. Wang, F. Wu - *On the vulnerabilities and Protection of OSPF Protocol*. Proceedings of 1998 International Conference on Computer Communications and Networks, 1998.

[9] T. Clausen and P. Jacquet - *Optimized Link State Routing Protocol (OLSR) - IETF RFC 3626*. October 2003.

[10] A. Tønnesen - *Implementing and extending the Optimized Link State Routing Protocol*. Master Degree Thesis, University of Oslo (UniK), 2004.

[11] A. Tønnesen, A. Hafslund and Ø. Kure - *The UniK - OLSR plugin library*. OLSR Interop and Workshop, 2004.