UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Campus de São José do Rio Preto

Rodrigo Martins da Conceição

Implementação e avaliação do protocolo μTESLA: proposta de uso nas redes veiculares

São José do Rio Preto
2014

Rodrigo Martins da Conceição

Implementação e avaliação do protocolo μTESLA: proposta de uso nas redes veiculares

Dissertação apresentada como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação, junto ao Programa de Pós-Graduação em Ciência da Computação, Área de Concentração – Computação Aplicada, do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", Campus de São José do Rio Preto.

Orientadora: Profª. Drª. Renata Spolon Lobato
Coorientador: Prof. Dr. Aleardo Manacero Junior

São José do Rio Preto
2014

Conceição, Rodrigo Martins da.
Implementação e avaliação do protocolo μTESLA : proposta de uso nas redes veiculares / Rodrigo Martins da Conceição. -- São José do Rio Preto, 2014
84 f. : il., gráfs.

Orientador: Renata Spolon Lobato
Coorientador: Aleardo Manacero Junior
Dissertação (mestrado) – Universidade Estadual Paulista "Júlio de Mesquita Filho", Instituto de Biociências, Letras e Ciências Exatas

1. Computação. 2. Tecnologia da informação – Sistemas de segurança. 3. Redes ad hoc veiculares (Redes de Computadores). 4. Radiodifusão. I. Lobato, Renata Spolon. II. Manacero Junior, Aleardo. III. Universidade Estadual Paulista "Júlio de Mesquita Filho". Instituto de Biociências, Letras e Ciências Exatas. IV. Título.

CDU – 681.3.025

Ficha catalográfica elaborada pela Biblioteca do IBILCE
UNESP – Câmpus de São José do Rio Preto

Rodrigo Martins da Conceição

Implementação e avaliação do protocolo μTESLA: proposta de uso nas redes veiculares

Dissertação apresentada como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação, junto ao Programa de Pós-Graduação em Ciência da Computação, Área de Concentração – Computação Aplicada, do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", Campus de São José do Rio Preto.

Banca Examinadora

Profª. Drª. Renata Spolon Lobato
UNESP – São José do Rio Preto
Orientadora

Prof. Dr. Norian Marranghello
UNESP – São José do Rio Preto

Prof. Dr. Hélio Crestana Guardia
UFSCar – São Carlos

São José do Rio Preto
05/dezembro/2014

Dedico este trabalho a toda minha família, que é minha fonte de inspiração e base da minha formação e educação como ser humano. Família que sempre esteve do meu lado, apoiando e incentivando, para que minha jornada não terminasse no meio do caminho.

Dedico, especialmente, ao meu avô Vicente. Homem simples, mas um exemplo de coragem, trabalho, respeito, honestidade e dignidade. Ele verá a conclusão de mais essa etapa de algum lugar ainda desconhecido por nós. Que o senhor me abençoe!

AGRADECIMENTOS

Agradeço à professora Renata pelo apoio e incentivo dispensados a mim, desde que comecei a cursar as primeiras disciplinas no programa de mestrado, ainda como aluno especial. Agradeço, ainda, pela paciência e compreensão com relação às minhas dificuldades e aos meus horários.

E ao professor Aleardo por toda ajuda e dedicação, principalmente na elaboração de artigos e durante as reuniões de grupo.

“Toda ciência seria supérflua se a aparência e a essência das coisas coincidissem diretamente.”

Karl Marx

RESUMO

Uma rede veicular – ou VANET (*Vehicular Ad Hoc Network*) – é um tipo especial de rede móvel na qual os nós são veículos ou estações base. Essas redes são formadas por sistemas de comunicação entre veículos automotores equipados com dispositivos de comunicação sem fio de curto alcance, ou entre um veículo e a infraestrutura fixa (estação base) localizada à margem da estrada. A partir dessa tecnologia é possível desenvolver diversas aplicações, classificadas em três áreas: segurança, informação e entretenimento. Este trabalho aborda o problema da segurança, com uma proposta de implementação e avaliação do µTESLA, um protocolo de autenticação usado para garantir a autenticidade da origem das mensagens enviadas por radiodifusão. Esse protocolo gera poucas sobrecargas de processamento e armazenamento, pois são utilizadas apenas primitivas simétricas e revelação tardia de chaves para efetivar a autenticação das mensagens, não necessitando de mecanismos mais complexos como assinatura digital ou criptografia assimétrica. Essa implementação foi testada e validada através do uso de dois simuladores gratuitos e de código aberto, a saber: VanetMobiSim, que forneceu os padrões de mobilidade de tráfego, e NS-2, que simulou a rede veicular e suas transferências de pacotes (mensagens).

Palavras-chave: µTESLA. Rede Veicular. Autenticação. Radiodifusão. Segurança de rede.

ABSTRACT

A vehicular network - or VANET (Vehicular Ad Hoc Network) - is a special type of mobile network in which nodes are vehicles or base stations. These networks are formed by communication systems between automotive vehicles equipped with short range wireless communication devices, or between a vehicle and the fixed infrastructure (base station) located on the roadside. From this technology it is possible to develop various applications, classified into three areas: safety, information and entertainment. This study addresses the security problem with a proposal for implementation and evaluation of the μTESLA, an authentication protocol used to guarantee the authenticity of the origin of messages sent in broadcasting. This protocol generates reduced processing and storage overloads, since only symmetric primitives and delayed disclosure of keys are used to carry out the authentication of messages, not requiring more complex mechanisms such as digital signature and asymmetric encryption. This implementation has been tested and validated through the use of two free and open source simulators, namely: VanetMobiSim, which was used to provide the traffic mobility patterns, and NS-2, which was use to simulate the vehicular network and their packet transfers (messages).

Keywords: μTESLA. Vehicular Network. Authentication. Broadcasting. Network security.

LISTA DE ILUSTRAÇÕES

Figura 1. Estação base enviando informações para os veículos. 15

Figura 2. Uma VANET. 23

Figura 3. Comunicação entre veículos. 24

Figura 4. Comunicação entre veículo e infraestrutura de beira de estrada. 25

Figura 5. Comunicação baseada em roteamento. 26

Figura 6. Aviso de local perigoso. 27

Figura 7. Aproximação de veículo de emergência. 28

Figura 8. Aviso de engarrafamento. 29

Figura 9. Visão simplificada do NS-2. 37

Figura 10. Arquitetura geral do NS-2. 37

Figura 11. Topologias da estrada disponíveis no VanetMobiSim. 41

Figura 12. Autenticação da mensagem. 44

Figura 13. Troca de mensagens. 46

Figura 14. Exemplo de funcionamento do μTESLA. 49

Figura 15. Diagrama de classe de MTeslaHeaderClass. 53

Figura 16. Diagrama de classe de Estacao_BaseClass e VeiculoClass. 53

Figura 17. Diagrama de classe de Estacao_Base e de Veiculo. 54

Figura 18. Diagrama de sequência das trocas de mensagens. 56

Figura 19. Pacote de dados com requisição de sincronização. 57

Figura 20. Pacote de dados para sincronização. 58

Figura 21. Pacote de dados contendo mensagem. 59

Figura 22. Pacote de dados contendo chave. 59

Figura 23. Diagrama de estado do protocolo μTESLA. 60

Figura 24. Diagrama de classe de CMAPHeaderClass. 62

Figura 25. Diagrama de classe de AutoridadeClass, RSUClass e VeiculoClass. 63

Figura 26. Diagrama de classe de Autoridade, RSU e Veiculo..... 63
Figura 27. Configuração de variáveis. 67
Figura 28. Definição da topologia..... 67
Figura 29. Criação do objeto GOD. 68
Figura 30. Configuração dos nós móveis. 68
Figura 31. Instanciação dos nós móveis. 69
Figura 32. Definição de posição dos nós móveis. 69
Figura 33. Criação de agentes e ligação com os nós. 69
Figura 34. Escalonador de eventos..... 70
Figura 35. Parâmetros de configuração do padrão 802.11p. 71
Figura 36. Intervalo de transmissão. 72
Figura 37. Tempo médio de recebimento de mensagens. 74
Figura 38. Perda de pacotes..... 75

LISTA DE ABREVIATURAS E SIGLAS

ACIRI	*AT&T Center for Internet Research at ICSI*
CAM	*Cooperative Authentication Message*
CanuMobiSim	*Communication in Ad Hoc Networks for Ubiquitous Computing Mobility Simulator*
CMAP	*Cooperative Message Authentication Protocol*
CONSER	*Collaborative Simulation for Education and Research*
CSP	*Communicating Sequential Processes*
DARPA	*Defense Advanced Research Projects Agency*
ECDSA	*Elliptic Curve Digital Signature Algorithm*
ECIES	*Elliptic Curve Integrated Encryption Scheme*
ERTICO	*European Road Transport Telematics Implementation Coordenation Organization*
EVITA	*E-safety Vehicle Intrusion Protected Applications*
GDF	*Geographic Data File*
GOD	*General Operations Director*
GPS	*Global Positioning System*
GSA	*Group-based Source Authentication*
HLPSL	*High Level Protocol Specification Language*
ICSI	*International Computer Science Institute*
IDM_LC	*Intelligent Driver Model with Lane Changes*
INCT-SEC	Instituto Nacional de Ciência e Tecnologia em Sistemas Embarcados Críticos
ITS	*Intelligent Transportation Systems*
IVC	*Inter-Vehicle Communication*
JARI	*Japan Automobile Research Institute*
LBNL	*Lawrence Berkeley National Laboratory*
MAC	*Message Authentication Code*
MANET	*Mobile Ad hoc Network*
NAM	*Network Animator*
NCTUns	*National Chiao Tung University Network Simulator*
NHTSA	*National Highway Traffic Safety Administration*

NSF	*National Science Foundation*
NS-2	*Network Simulator* – versão 2
NTP	*Network Time Protocol*
OTcl	Extensão de Tcl Orientada a Objeto
PDA	*Personal Digital Assistant*
QoS	Quality of Service
RBM	*Regular Broadcast Message*
RSU	*RoadSide Unit*
RTT	*Round Trip Time*
SAMAN	*Simulation Augmented by Measurement and Analysis for Networks*
SNEP	*Secure Network Encryption Protocol*
SPINS	*Security Protocols for Sensor Networks*
Tcl	*Tool Command Language*
TclCL	Interface de Tcl com C++
TESLA	*Timed Efficient Stream Loss-tolerant Authentication*
TIGER	*Topologically Integrated Geographic Encoding and Referencing*
TraNS	*Traffic and Network Simulation*
UCB	*University of California campus at Berkeley*
USC/ISI	*University of Southern California/Information Sciences Institute*
UUID	*Universally Unique Identifier*
VANET	*Vehicular Ad hoc Network*
VanetMobiSim	*Veicular Ad Hoc Networks Mobility Simulator*
VINT	*Virtual InterNetwork Testbed*
VRC	V*ehicle-to-Roadside Communication*
VSPT	VANET *authentication with Signatures and Prediction-based* TESLA
V2I	V*ehicle-to-Infrastructure*
V2V	*Vehicle-to-Vehicle*
WAVE	*Wireless Access in the Vehicular Environment*
µTESLA	Micro TESLA
2LXORC	*2-Level eXclusive-Or Chain*

SUMÁRIO

1 INTRODUÇÃO 14
1.1 Introdução 14
1.2 Motivação 16
1.3 Objetivos 16
1.4 Trabalhos Relacionados 17
1.5 Organização da Dissertação 21
2 REDES *AD HOC* VEICULARES 22
2.1 Caracterização 22
2.2 Comunicação 24
2.3 Aplicações 26
2.4 Segurança 29
2.5 Desafios Relacionados às VANETs 30
2.6 Considerações Finais 32
3 FERRAMENTAS DE SIMULAÇÃO 34
3.1 Importância das Simulações 34
3.2 *Network Simulator* Versão 2 (NS-2) 35
3.2.1 Origem do NS-2 36
3.2.2 Arquitetura 36
3.3 VanetMobiSim 38
3.3.1 Características de Macromobilidade 40
3.3.2 Características de Micromobilidade 42
3.4 Considerações Finais 42
4 PROTOCOLO µTESLA 43
4.1 Autenticação de Mensagens 43
4.2 Legado do Protocolo TESLA 44
4.3 Descrição do Protocolo µTESLA 45
4.3.1 Transmissão de Mensagens Autenticadas 48
4.3.2 Exemplo de Funcionamento 49
4.4 Considerações Finais 50
5 DESENVOLVIMENTO E IMPLEMENTAÇÃO 51

5.1 Detalhes da Implementação do Protocolo µTESLA 51
5.2 Especificação do Protocolo µTESLA..... 57
5.3 Descrição do Protocolo CMAP 61
5.4 Considerações Finais..... 64
6 RESULTADOS.....65
6.1 Configurações do Cenário de Simulação 65
6.1.1 Simulação no VanetMobiSim 65
6.1.2 Simulação no NS-2 66
6.2 Viabilidade do Protocolo µTESLA..... 71
6.3 Comparação entre os Protocolos µTESLA e CMAP 73
6.4 Considerações Finais..... 75
7 CONCLUSÃO 77
7.1 Análise dos Resultados..... 77
7.2 Dificuldades Encontradas..... 78
7.3 Sugestões para Trabalhos Futuros 78
REFERÊNCIAS 80
APÊNDICE A – Tabelas com os dados das simulações.....84

1 INTRODUÇÃO

Neste capítulo, faz-se uma apresentação da dissertação. Na seção 1.1 é apresentada uma introdução ao tema das redes veiculares. Na seção 1.2 é feita uma explicação a respeito da escolha do tema proposto. Na seção 1.3 são descritos os objetivos almejados e o problema que este trabalho pretende abordar. Na seção 1.4 são descritos alguns dos principais estudos e projetos existentes relacionados com este trabalho. Na seção 1.5 é explicado de que forma esta dissertação foi organizada.

1.1 Introdução

Os sistemas de transporte eficiente têm gerado grande impacto no crescimento econômico de alguns países. Essa nova tecnologia tem auxiliado no aumento de recursos de mobilidade para cidadãos, sociedades e economias, além de gerar uma integração mais ampla para toda a sociedade por meio das inovações tecnológicas no setor de transporte (RAFIQ et al., 2013).

Esses meios de transporte inteligente são os denominados ITS (*Intelligent Transportation Systems*), que, incorporando características de segurança, navegação e entretenimento, são considerados requisitos básicos para a riqueza sustentável e a prosperidade das sociedades modernas (RAFIQ et al., 2013).

Nos ITS, cada veículo tem o papel de emissor, receptor e roteador de informações. Para que possa ocorrer uma comunicação entre os próprios veículos ou entre veículo e infraestrutura à margem da estrada (estação base ou RSU – *RoadSide Unit*), é necessário que os veículos estejam equipados com algum tipo de interface de rádio (OBU – *OnBoard Unit*) que habilita uma rede sem fio de curto alcance.

As RSUs são dispositivos estáticos que guardam aplicações e fornecem serviços, enquanto as OBUs executam as aplicações para usar um serviço específico. Além disso, os veículos também precisam de um equipamento que forneça informação de posição, como o GPS (*Global Positioning System*) (ZEADALLY et al., 2010).

Na Figura 1 é mostrado um cenário real no qual uma estação base envia informações para os veículos que trafegam nas imediações.

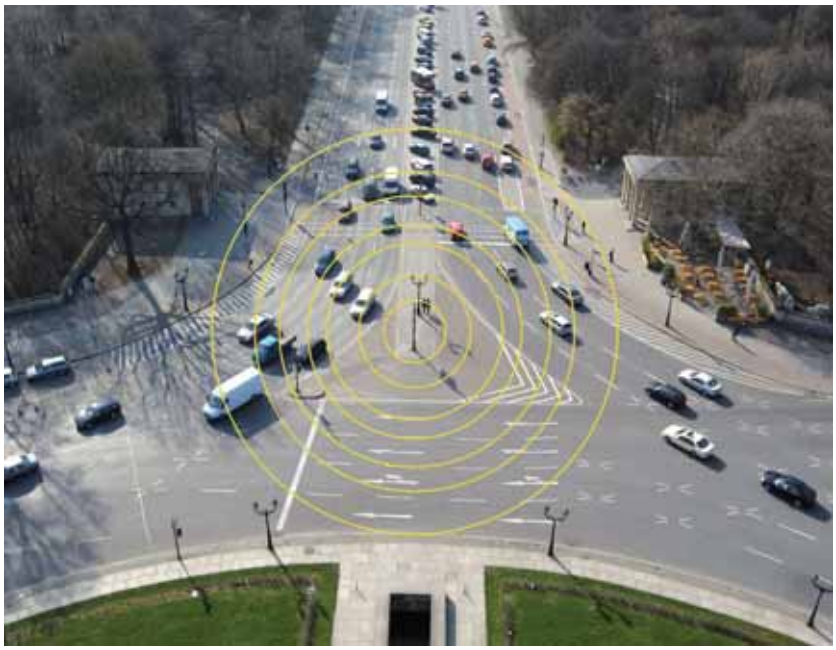


Figura 1. Estação base enviando informações para os veículos (RAFIQ et al., 2013, p. 45).

O conceito de VANET (*Vehicular Ad hoc Network*) é uma abordagem mais recente que está sendo objeto de diversas pesquisas e projetos na área dos ITS.

Utilizando-se de uma rede *ad hoc* móvel, essa rede veicular permite uma comunicação direta entre veículos, na qual dois veículos se comunicam sem a interferência de um terceiro; comunicação entre um veículo e uma estação base, na qual veículo e estação base se comunicam diretamente; ou uma comunicação baseada em roteamento, na qual uma mensagem passa por um ou mais veículos até chegar ao veículo de destino (ZEADALLY et al., 2010).

Embora essa comunicação forneça inúmeros benefícios para os motoristas, uma simples troca de informação não é suficiente para que uma comunicação seja bem sucedida. Para que as VANETs funcionem de forma adequada é preciso que se estabeleça uma comunicação eficiente e segura.

Essas qualidades podem ser alcançadas por meio de um protocolo de segurança, usado durante as trocas de mensagens. Um mecanismo utilizado para esse fim é o protocolo de autenticação de mensagens μTESLA, que será discutido no capítulo 4.

1.2 Motivação

As redes veiculares constituem um campo de pesquisa emergente, sendo consideradas essenciais para a cooperação entre veículos na estrada.

Seus modelos de comunicação podem ser usados, por exemplo, para a melhoria da segurança de tráfego, para o planejamento de melhores rotas, para o controle de congestionamento de tráfego, para a transmissão de avisos de emergência, das condições das estradas e de informações de entretenimento ao motorista (MARTINEZ et al., 2009; RAYA, PAPADIMITRATOS, HUBAUX, 2006).

Com os benefícios esperados pelas aplicações em VANETs, bem como com os esforços dispensados para a pesquisa por meio de indústrias e de instituições acadêmicas e financiamentos dos governos, vislumbra-se que em um futuro próximo grande parte da frota automotiva esteja equipada com dispositivos capazes de se comunicarem.

Contudo, as redes veiculares ainda apresentam vários problemas e desafios, mostrando-se um campo de pesquisa bastante promissor. Alguns desses problemas já foram identificados e necessitam de mais estudos, destacando-se a questão da segurança.

Segundo Zeadally et al. (2010), um desafio chave que permanece nas VANETs, no que tange à segurança, é fornecer autenticação do emissor de uma mensagem nos cenários de comunicação por radiodifusão. Além disso, acrescentam que a radiodifusão é uma área de pesquisa forte nas VANETs, devido ao número significativo de mensagens transmitidas dessa forma.

Esse desafio de transmitir, por radiodifusão, mensagens autenticadas levou à escolha do tema, que é a implementação e a avaliação de um protocolo de autenticação de mensagens enviadas por radiodifusão, o protocolo µTESLA (Micro *Timed Efficient Stream Loss-tolerant Authentication*) (PERRIG et al., 2001).

1.3 Objetivos

µTESLA é um protocolo de autenticação de mensagens, projetado especificamente para as redes de sensores, nas quais os recursos de memória, de processamento e de energia são limitados.

Ao se utilizar esse protocolo para realizar as trocas de mensagens com diversos receptores, garante-se a autenticidade da origem da mensagem. Ou seja, o receptor sabe que a mensagem enviada, supostamente, pelo emissor *A* realmente partiu de *A*. Esse recurso evita, entre outras coisas, a situação em que o emissor de uma mensagem maliciosa ou enganosa se faz passar por outra entidade a fim de enganar o receptor daquela mensagem.

A proposta do presente trabalho é a implementação e a avaliação desse protocolo para que seja feita a autenticação de mensagens enviadas por radiodifusão nas VANETs. Essa implementação foi feita em linguagem de programação C++, a fim de que fosse possível agregá-la como um novo módulo ao simulador de rede NS-2 - *Network Simulator* versão 2 (THE NETWORK SIMULATOR, [2014?]).

Tal protocolo foi testado e validado através do próprio NS-2, que utilizou um modelo de mobilidade gerado pelo simulador de tráfego VanetMobiSim (HÄRRI; FIORE, 2005/2006) como parâmetro de mobilidade para os nós. Ou seja, os padrões gerados pelo simulador VanetMobiSim serviram como dados de entrada para a simulação no NS-2.

Estes dois simuladores são disponibilizados gratuitamente e permitem que seus códigos fontes sejam alterados. Essa característica é importante para que se possa adicionar a implementação do protocolo μTESLA às funcionalidades já presentes no simulador.

Além disso, foi realizada uma comparação entre os protocolos μTESLA e CMAP (*Cooperative Message Authentication Protocol*) (HAO et al., 2011). Esses dois protocolos foram simulados no NS-2 e seus resultados foram analisados e comparados, mostrando a viabilidade do uso do protocolo μTESLA nas redes veiculares.

1.4 Trabalhos Relacionados

Com a promessa de melhorar a segurança nas estradas e fornecer maior conforto aos motoristas e passageiros, as redes veiculares vêm atraindo a atenção dos setores acadêmico, governamental e industrial. Como consequência, diversos projetos importantes foram desenvolvidos ou ainda estão em desenvolvimento em diferentes partes do mundo.

Nos Estados Unidos, a ITS America é uma organização dedicada ao avanço da pesquisa, do desenvolvimento e da implantação de Sistemas Inteligentes de Transporte. Ela

conta com a participação de órgãos públicos, empresas privadas e instituições acadêmicas e de pesquisa (ITS AMERICA, [2014?]).

ITS JPO (*Intelligent Transportation Systems Joint Program Office*) é um programa que mantém o foco nos veículos e infraestrutura inteligentes, bem como na criação de um sistema inteligente de integração entre esses dois componentes. O programa investe em pesquisas e implantação, incluindo o apoio em tecnologia e treinamento (RESEARCH AND INNOVATIVE TECHNOLOGY ADMINISTRATION, 2013).

Na União Europeia, destaca-se a ERTICO (*European Road Transport Telematics Implementation Coordenation Organization)*, uma organização fundada pela união entre Comissão Europeia, ministérios de transporte e indústrias europeias. A ERTICO é uma rede de associados interessados na melhoria e na padronização de serviços de transporte e ITS na Europa (ERTICO ITS EUROPE, [2014?]).

Car-to-Car Communications Consortium visa contribuir para o melhoramento da segurança e eficiência do tráfego rodoviário, por meio do princípio dos Sistemas de Transporte Inteligentes cooperativos. Esse consórcio apoia a criação de um padrão europeu para a comunicação de veículos de todas as marcas (CAR 2 CAR COMMUNICATION CONSORTIUM, [2014?]).

EVITA é um projeto que objetiva desenvolver uma arquitetura de segurança para redes *on-board* automotivas, na qual todos os componentes e informações de segurança serão protegidos contra adulterações e manipulações maliciosas (EVITA, [2008?]).

No Japão, destaca-se o JARI (*Japan Automobile Research Institute*), uma organização de interesse público destinada às atividades de pesquisas e testes automotivos. O JARI avalia diversas tecnologias envolvendo veículos inteligentes, prevenção de colisões e sistemas de segurança (JAPAN AUTOMOBILE RESEARCH INSTITUTE, [2014?]).

No Brasil, o Instituto Nacional de Ciência e Tecnologia em Sistemas Embarcados Críticos (INCT-SEC) possui um grupo de trabalho que desenvolve sistemas de navegação autônoma e assistida para veículos terrestres. O sistema de controle autônomo – utilizando-se da integração de múltiplos sensores, como câmeras, GPS, bússola digital e outros – é capaz de conduzir um veículo sem qualquer intervenção humana ou, então, notificar o motorista de uma possível situação de risco.

Com o objetivo de reduzir o número de acidentes em ruas e rodovias e aumentar a eficiência no trânsito, uma aplicação que está se destacando é o projeto CARINA (Carro Robótico Inteligente para Navegação Autônoma). Esse projeto consta de dois veículos automatizados e equipados para navegação autônoma, sendo que o segundo (CARINA 2) foi instrumento do primeiro teste realizado em ambientes urbanos na América Latina (INSTITUTO NACIONAL DE CIÊNCIA E TECNOLOGIA EM SISTEMAS EMBARCADOS CRÍTICOS, [2014?]).

Além desses projetos mais abrangentes, diversos outros trabalhos mais específicos estão sendo publicados atualmente. Segue uma breve descrição de alguns dos trabalhos referentes à autenticação de mensagens nas redes veiculares.

Lin e Li (2013) propõem um esquema cooperativo de autenticação de mensagens para VANETs, prometendo diminuir a sobrecarga e o atraso na autenticação dos veículos eliminando redundâncias.

Na mesma linha, Hao et al. (2011) propõem um protocolo de autenticação cooperativa de mensagem (CMAP), com o objetivo de aliviar a carga computacional dos veículos. Nesse sistema, alguns veículos são selecionados, de acordo com suas posições geográficas, para serem os verificadores. Caso uma mensagem recebida por um verificador seja verdadeira, ela é autenticada; caso contrário, ela é rejeitada e o verificador envia uma mensagem de aviso para os outros veículos. Se a mensagem for recebida por um veículo que não é um verificador, ela é armazenada e o veículo espera pela mensagem de aviso do verificador.

Perrig et al. (2000) apresentam o TESLA (*Timed Efficient Stream Loss-tolerant Authentication*), um protocolo de autenticação de transmissão por radiodifusão eficiente, com pouca sobrecarga de comunicação e computação, escalável e tolerante à perda de pacotes.

Lyu et al. (2013) propõem um novo mecanismo de autenticação VANET, denominado VSPT (VANET *authentication with Signatures and Prediction-based TESLA*). A ideia desse protocolo é combinar as vantagens da assinatura digital (ECDSA – *Elliptic Curve Digital Signature Algorithm*) e do protocolo *TESLA*.

GSA (*Group-based Source Authentication*) é um protocolo desenvolvido por Lu et al. (2010) para autenticação de mensagem na origem. O protocolo usa os atributos de grupo

para proteger a transmissão de dados em uma comunicação intragrupo e utiliza o esquema do protocolo TESLA para realizar autenticação na origem em uma comunicação intergrupo.

Studer et al. (2008) propõem uma versão modificada do TESLA para autenticação em VANETs. O TESLA++ fornece autenticação de transmissão por radiodifusão eficiente como o TESLA, porém com exigências de memória reduzidas.

Perrig et al. (2001) desenvolveram um conjunto de protocolos de segurança para ambientes com recursos limitados e comunicação sem fio (para redes de sensores mais especificamente). Nesse projeto está o protocolo μTESLA, uma versão adaptada do protocolo TESLA, que fornece transmissão por radiodifusão autenticada em ambientes de recursos limitados.

A proposta de Yeo e Youm (2010) é baseada no protocolo μTESLA. O 2LXORC (*2-Level eXclusive-Or Chain*) utiliza uma cadeia de XOR em dois níveis, garantindo tolerância à perda de dados em longo prazo e suporte para vários emissores.

Jaballah et al. (2011) verificam e modelam formalmente três protocolos de autenticação (*Multi-Level*-μTESLA, *Staggered Multi-Level*-μTESLA e *Bloom Filter*) usando a ferramenta de verificação de modelo Avispa e a linguagem HLPSL (*High Level Protocol Specification Language*). Os resultados desse estudo mostram que os protocolos podem ser usados com segurança, entretanto, apresentam alguns problemas de ataque de negação de serviço.

Por fim, Wang et al. (2011) fazem uma análise do protocolo μTESLA empregando uma modelagem que utiliza um processo formal de álgebra em CSP (*Communicating Sequential Processes*). De acordo com os autores, a realização de testes com ataques à rede mostraram bons resultados, demonstrando que a propriedade de autenticação de radiodifusão do protocolo é correta e satisfatória.

Esses trabalhos representam apenas uma pequena parcela de um vasto conjunto, ilustrando o quanto é importante e promissor esse campo de pesquisa. Apesar dos inúmeros trabalhos já realizados, ainda existem muitas questões em aberto, como por exemplo, a implementação de esquemas de autenticação e troca de mensagens mais eficiente e segura, para as transmissões por radiodifusão.

Com esta dissertação, pretendemos gerar alguma contribuição para as pesquisas da área de segurança das redes veiculares, uma vez que estamos usando um protocolo no qual

alguns autores, como Yeo e Youm (2010) e Jaballah et al. (2011), basearam-se para desenvolver novas propostas e outros autores, como Wang et al. (2001), analisaram e aprovaram μTESLA, após testes com ataques à rede.

A novidade da nossa proposta é a utilização do protocolo μTESLA nas redes veiculares, com implementação, simulação e análise de resultados, o que reforça ainda mais a relevância do trabalho, visto que não encontramos na literatura atual outra proposta dessa natureza.

1.5 Organização da Dissertação

Os demais capítulos deste trabalho foram organizados conforme segue.

No capítulo 2, é feita uma contextualização a respeito das VANETs, explicando conceitos, características, tipos de comunicação, aplicações possíveis, questões sobre segurança e os principais desafios relacionados a esse tipo de rede.

No capítulo 3, são explicados os objetivos e a importância das ferramentas de simulação em VANETs. Além disso, são descritos os dois simuladores usados neste trabalho, a saber, o simulador de rede NS-2 e o simulador de tráfego VanetMobiSim.

No capítulo 4, há uma introdução a respeito de autenticação de mensagens, uma breve comparação entre os protocolos TESLA e μTESLA e uma explanação mais detalhada do protocolo μTESLA, detalhando seu funcionamento.

No capítulo 5, são apresentados os detalhes sobre o desenvolvimento e a implementação do protocolo μTESLA, incluindo sua especificação, a definição dos pacotes de dados e seu diagrama de estados.

No capítulo 6, são mostrados os resultados obtidos através das simulações no NS-2. São exibidos as configurações do cenário de simulação, os testes com resultados sobre a viabilidade do μTESLA e uma comparação entre os protocolos μTESLA e CMAP.

Por fim, no capítulo 7, são apresentadas a análise dos resultados, as dificuldades encontradas no decorrer do trabalho e as sugestões para trabalhos futuros.

2 REDES *AD HOC* VEICULARES

Neste capítulo são abordados os principais itens referentes às redes *ad hoc* veiculares (VANETs). Na seção 2.1 são descritas as características desse tipo de rede. Na seção 2.2 são apresentados os três tipos de comunicação mais comuns. Na seção 2.3 são ilustradas aplicações com alguns casos de uso. Na seção 2.4 são descritos alguns pontos importantes relacionados à segurança. Na seção 2.5 são apontados os principais desafios relacionados à pesquisa sobre as VANETs. Na seção 2.6 são apresentadas as considerações finais do capítulo.

2.1 Caracterização

Para entender melhor o que são as VANETs, primeiro é preciso conhecer as MANETs ou *Mobile Ad Hoc Networks*. Estas são redes compostas de um conjunto de dispositivos de comunicação móveis, capazes de se interconectar espontaneamente (ponto a ponto) sem qualquer infraestrutura preexistente.

A importância das MANETs em aplicações militares e nos negócios é crescente. A ampla gama de dispositivos móveis leves e de baixo custo – como os telefones celulares, os PDAs (*Personal Digital Assistant*), os *tablets* e os *smartphones* – que incorporam *Bluetooth* e Wi-Fi, permite a criação espontânea de sub-redes em pontos importantes pela cidade.

Essas redes poderiam, então, constituir a infraestrutura de inúmeras aplicações, tais como sistemas de emergência e de cuidados com a saúde, aplicativos colaborativos, jogos, propagandas, dentre outras aplicações (HOGIE; BOUVRY; GUINAND, 2006).

As VANETs, por sua vez, são um subconjunto das MANETs, nas quais os nós são veículos que se deslocam rapidamente e com regularidade determinada pelo tráfego. Essa tecnologia permite a comunicação entre veículos e também entre um veículo e a infraestrutura próxima à estrada, através de um dispositivo sem fio instalado nos veículos (HASSAN, 2009).

A partir daí, novas oportunidades e tecnologias relacionadas começam a aparecer, utilizando esse conhecimento em aplicações como controle de congestionamento de tráfego e de acidentes nas rodovias, com o intuito de aumentar a segurança dos veículos nas estradas, melhorar a eficiência do tráfego e o conforto para os motoristas e passageiros.

Por fim, existem algumas características que comumente estão presentes nas VANETs. Hassan (2009) elenca algumas delas:

- Os nós de uma VANET são os veículos e as unidades de beira de estrada;
- O movimento dos veículos é muito rápido;
- Os padrões de deslocamento são limitados pela topologia da estrada;
- Os veículos atuam como transceptores, ou seja, enviando e recebendo mensagens ao mesmo tempo;
- A densidade de veículos varia de tempos em tempos, por exemplo, podendo aumentar durante o horário de pico e diminuir à noite.

Na Figura 2, é ilustrada uma típica VANET com veículos, estações base e possíveis tipos de comunicação. É possível observar as trocas de mensagens de segurança que podem ocorrer, por exemplo, após um evento de emergência, fornecendo melhores condições de reação aos motoristas.

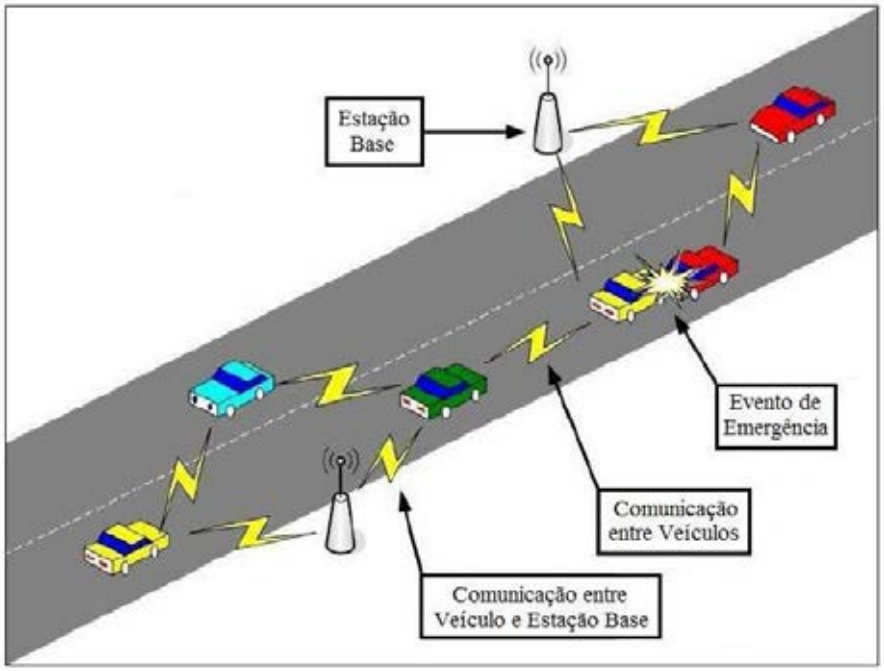


Figura 2. Uma VANET (RAYA; HUBAUX, 2007, p. 42).

Essa troca de informação tem que acontecer de forma eficiente – visto que os veículos podem trafegar em altas velocidades e se deslocarem por todas as direções – e segura – com garantias de autenticidade de origem e integridade de conteúdo.

2.2 Comunicação

No contexto da comunicação, cada veículo da rede assume o papel de emissor, receptor e roteador para transmitir informações para toda a rede veicular ou para uma agência de transporte, que usa essas informações para aumentar a segurança de motoristas e passageiros e melhorar o tráfego, deixando-o com fluxo mais livre.

Zeadally et al. (2010) apresentam as três configurações mais comuns de comunicação em VANETs, descritas a seguir.

Comunicação entre Veículos

A comunicação entre veículos, também denominada V2V (*vehicle-to-vehicle*) ou ainda IVC (*inter-vehicle communication*), é ilustrada na Figura 3. Esse tipo de comunicação usa uma transmissão por multissaltos para enviar informações relacionadas ao tráfego para um grupo de receptores (ou todos os receptores).

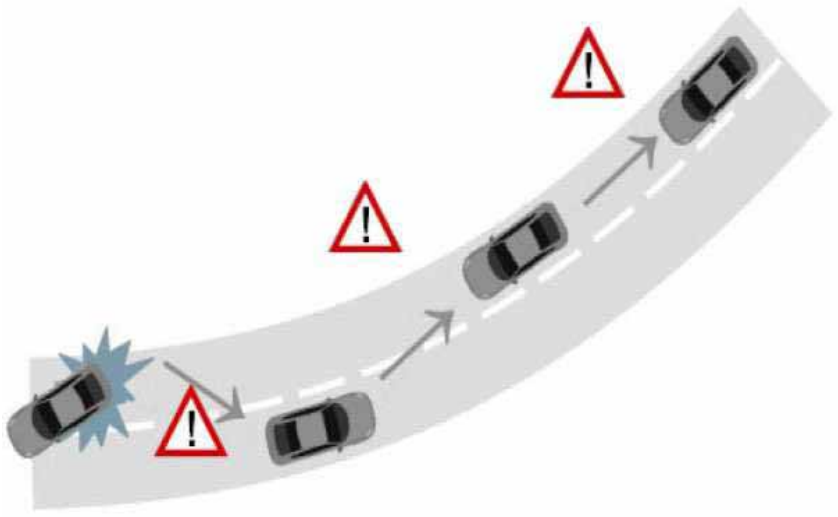


Figura 3. Comunicação entre veículos (ZEADALLY et al., 2010, p. 218).

Dois tipos de encaminhamento de mensagens são comuns: difusão simples e difusão inteligente. Em ambos os casos, se uma mensagem recebida foi enviada por um veículo que está mais atrás, ela é simplesmente ignorada; mas se a mensagem foi enviada por um veículo que está mais à frente, ela é retransmitida em uma nova difusão, garantindo que todos os veículos recebam a mensagem.

A diferença entre as duas abordagens é que na difusão simples os veículos enviam as mensagens periodicamente e em intervalos regulares de tempo, enquanto na difusão

inteligente o número de transmissões de mensagens sobre um dado evento de emergência é limitado.

Neste caso, quando um veículo detecta o recebimento de uma mensagem emitida por um veículo posicionado mais atrás, ele assume que pelo menos um veículo de trás recebeu a mensagem e encerra esse ciclo de difusões, pois este veículo ficará encarregado de retransmitir a mensagem para os outros veículos. Caso um veículo receba uma mesma mensagem de mais de uma origem, somente a primeira será considerada.

Comunicação entre Veículo e Infraestrutura

Nesse modo de comunicação, também chamada de V2I (*vehicle-to-infrastructure*) ou VRC (*vehicle-to-roadside communication*), as mensagens são enviadas pelas estações base para todos os veículos equipados na vizinhança, fazendo uma transmissão de único salto, conforme mostrado na Figura 4.



Figura 4. Comunicação entre veículo e infraestrutura de beira de estrada (ZEADALLY et al., 2010, p. 219).

Uma conexão de banda larga é fornecida entre os veículos e unidades de estrada que, colocada a cada quilômetro ou menos, permite transmitir altas taxas de dados. Com isso, é possível, por exemplo, gerar e enviar uma mensagem de alerta para um veículo que exceda o limite de velocidade da estrada, estabelecido pela unidade de beira de estrada conforme um calendário ou condições de tráfego.

Comunicação Baseada em Roteamento

Esse tipo de configuração baseia-se em uma transmissão de múltiplos saltos. Por exemplo, no caso de um veículo requisitar uma informação que somente um veículo específico possui, a mensagem de requisição é propagada de veículo a veículo até que seja alcançado o veículo desejado.

Na Figura 5 é ilustrada uma situação em que o veículo *A* precisa de uma informação que somente o veículo *B* possui.

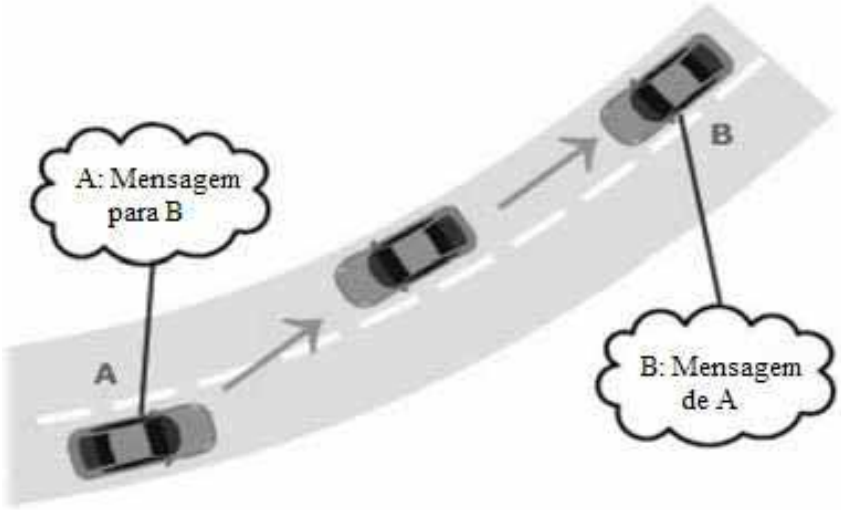


Figura 5. Comunicação baseada em roteamento (ZEADALLY et al., 2010, p. 219).

Conforme mostrado na Figura 5, o veículo *A* envia uma mensagem solicitando uma informação desejada. Um veículo intermediário recebe esse pedido e retransmite a mensagem, pois não possui a informação solicitada por *A*. O veículo *B* recebe a mensagem e, sabendo que possui a informação solicitada por *A*, envia, imediatamente, outra mensagem contendo a informação para o veículo que recebeu o pedido anteriormente. Este, por sua vez, fica encarregado de retransmitir a informação para o veículo que fez o pedido de origem, isto é, o veículo *A*.

2.3 Aplicações

O Consórcio de Comunicação *CAR 2 CAR* é uma organização industrial sem fins lucrativos, iniciada por fabricantes de veículos europeus e apoiados por fornecedores de equipamentos, organizações de pesquisa e outros. Esse grupo dedica suas pesquisas ao

melhoramento da segurança e da eficiência do tráfego rodoviário e faz uma classificação da evolução das aplicações em VANETs em três principais áreas (CAR 2 CAR COMMUNICATION CONSORTIUM, [2014?]):

- Assistência ao motorista, aumentando a segurança na estrada e diminuindo o número de acidentes;
- Aumento da eficiência do tráfego com controle de congestionamento, resultando em redução do consumo de combustível e tempo;
- Fornecimento de serviços de informação e comunicações a usuários, oferecendo aplicações de conforto e de negócios para motoristas e passageiros.

A seguir são apresentados alguns casos de uso apontados pelo grupo *CAR 2 CAR*.

Aviso de Local Perigoso

Um veículo detecta um local perigoso (uma mancha de óleo na pista, por exemplo) analisando as informações do sensor instalado no veículo. Este transmite esta informação com a localização do perigo aos veículos ao seu redor. Uma motocicleta que se aproxima recebe este aviso, porém, como o local não está em sua própria rota, apenas armazena as informações e começa a emitir a informação também. Essa situação é ilustrada na Figura 6.



Figura 6. Aviso de local perigoso (CAR 2 CAR COMMUNICATION CONSORTIUM, [2014?]).

Pouco tempo depois, a moto fornece a informação para um carro que está prestes a passar pelo óleo e o condutor deste carro pode diminuir a velocidade para passar no local perigoso com segurança.

Aproximação de um Veículo de Emergência

Quando se trata de situações que afetam a segurança de vidas cada segundo é crucial. Sendo assim, os motoristas devem abrir caminho para um veículo de emergência, como ambulância ou carro de polícia, para agilizar sua passagem.

Na Figura 7 é ilustrada uma situação em que os motoristas encostam seus veículos, para que uma viatura de bombeiros consiga passar e seguir seu caminho o mais rápido possível.

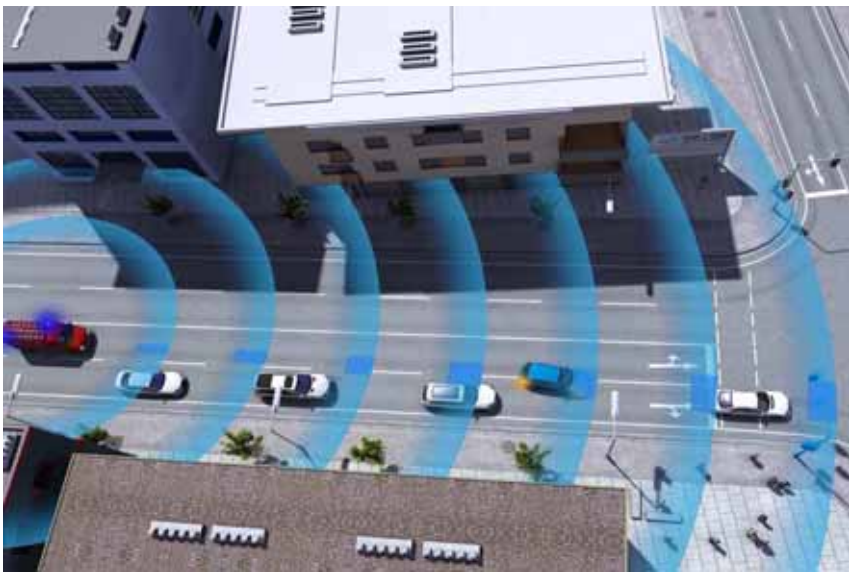


Figura 7. Aproximação de veículo de emergência (CAR 2 CAR COMMUNICATION CONSORTIUM, [2014?]).

Ao ouvir a sirene, nem sempre o motorista tem informações sobre a localização exata do veículo de emergência. Por isso, o aviso de aproximação pode ajudar esse motorista a reagir adequadamente, mesmo quando a sirene e as luzes ainda não estiverem audíveis ou visíveis.

Evitando Engarrafamentos

Um sistema pode coletar informações sobre a situação do trânsito local, como no caso de um acidente seguido de engarrafamento, por exemplo, em que um veículo reúne informações a respeito da velocidade média para todas as vias da estrada. Em seguida, transmite essa informação para outros veículos que estejam em uma área local de 50 a 100 km, conforme ilustrado na Figura 8.

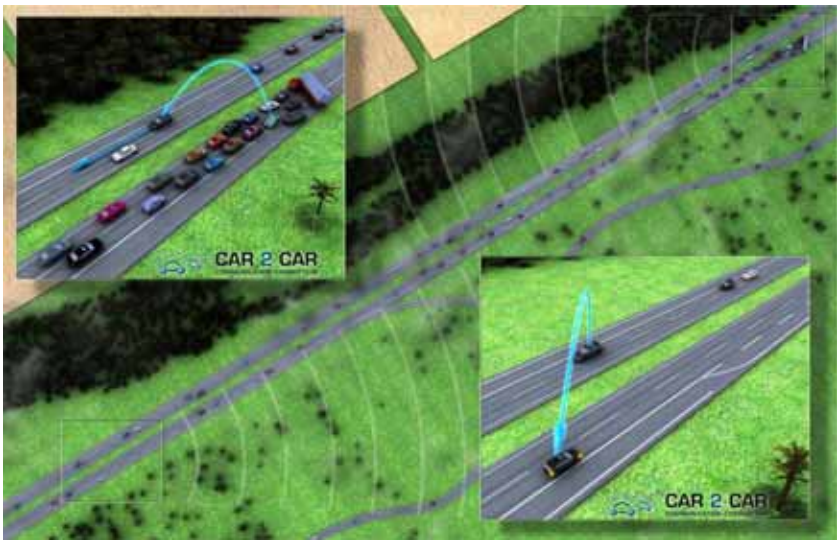


Figura 8. Aviso de engarrafamento (CAR 2 CAR COMMUNICATION CONSORTIUM, [2014?]).

Ao analisar a velocidade média de outros veículos, pode-se detectar um obstáculo ou um engarrafamento, podendo o motorista decidir por diminuir a velocidade ou seguir por outro caminho, se for possível.

2.4 Segurança

Raya e Hubaux (2007) dividem as aplicações direcionadas às redes veiculares em dois grupos importantes:

1. Aplicações relacionadas à segurança: essa categoria preocupa-se com situações críticas, onde a existência do serviço pode prevenir acidentes e preservar vidas;
2. Outras aplicações: relacionadas à otimização de tráfego e serviços de pagamento, localização e entretenimento.

Esses autores focam a pesquisa somente na primeira categoria por ser mais específica para o domínio automotivo e por ter problemas mais desafiadores.

Nesse contexto, Raya e Hubaux (2007) classificam as mensagens de segurança em três classes:

1. Informação de tráfego: usadas para disseminar as condições de tráfego em uma determinada região, sem restrições de tempo crítico;

2. Relacionadas à segurança em geral: utilizadas por aplicações de segurança pública, como prevenção de colisões, não podendo sofrer atrasos na entrega;

3. Relacionadas à responsabilidade: as mensagens são trocadas em caso de acidentes, portanto, a responsabilidade do remetente deve ser determinada, revelando sua identidade às autoridades policiais.

Uma propriedade comum de todas as classes de mensagens, garantem Raya e Hubaux (2007), é que elas são *geocast*, isto é, as informações de localização geográficas dos nós são utilizadas para que as mensagens sejam direcionadas e roteadas corretamente. Outra propriedade é que as mensagens são autônomas, ou seja, não existe nenhuma dependência de conteúdos entre elas.

Raya e Hubaux (2007), preocupados apenas com as redes veiculares, identificam alguns ataques realizados no momento da troca de mensagens, não considerando os ataques a veículos, como é o caso da segurança física de produtos eletrônicos para veículos.

Tais ataques são discriminados como segue:

- Informações falsas: os atacantes fornecem informações erradas à rede, a fim de afetar o comportamento de outros motoristas;
- Informações erradas do sensor: os atacantes alteram a sua posição, velocidade ou direção, a fim de escapar de suas responsabilidades, por exemplo, em um acidente;
- Divulgação da identificação de outros veículos: o atacante consegue rastrear a localização dos veículos, monitorando suas trajetórias e usando esses dados para uma variedade de propósitos;
- Negação de serviço: o atacante bloqueia canais ou “inunda” a rede com mensagens falsas, com o intuito de “derrubar” a VANET ou causar um acidente;
- Mascaramento: o atacante finge ser outro veículo usando identidades falsas.

Combinações desses ataques também são possíveis.

2.5 Desafios Relacionados às VANETs

Zeadally et al. (2010) discutem sobre alguns temas que ainda requerem mais investigação e que, portanto, representam desafios importantes no que se refere às VANETs. Esses temas são descritos na sequência.

Protocolos de Roteamento

Um assunto de fundamental importância em VANETs é o roteamento, já que a alta velocidade de mobilidade dos veículos e a rápida mudança na topologia tornam os protocolos de roteamento convencionais das MANETs inadequados para o ambiente das VANETs.

Essa situação levou os pesquisadores a buscarem algoritmos de roteamento escaláveis robustos para lidar com os problemas causados pela mobilidade dos veículos. Espera-se que abordagens novas e inovadoras possam oferecer melhor rendimento, diminuindo a perda de entrega de pacotes.

***Frameworks* de Segurança**

Suporte eficiente à segurança é um requisito importante para as VANETs. Nesse sentido, ainda existem vários desafios que precisam ser abordados nas áreas de autenticidade, confidencialidade dos dados e disponibilidade, por exemplo.

São necessárias mais pesquisas em *frameworks* de autenticação (que confirmam a autenticidade da origem das mensagens) escaláveis e capazes de proteger os veículos de possíveis atacantes.

Esses *frameworks* impedem que alguém se infiltre na rede usando identidade falsa, revelam sinais falsos de GPS, impedem a introdução de informação falsa na rede veicular e identificam ataques que suprimem, fabricam, alteram ou reproduzem mensagens legítimas.

O protocolo μTESLA, segundo Perrig et al. (2001), pretende resolver esse problema de autenticação, transmitindo mensagens por radiodifusão de forma eficiente, com baixa sobrecarga na rede e podendo ser escalado para um número grande de receptores.

Ainda segundo Perrig et al. (2001), o diferencial desse protocolo está na capacidade de transmitir, por radiodifusão, mensagens autenticadas utilizando somente primitivas de criptografia simétrica, ao passo que, em geral, os protocolos similares utilizam criptografia assimétrica. Perrig et al. (2001) explicam que isso é possível pela divulgação tardia das chaves responsáveis pelas autenticações das mensagens.

Qualidade de Serviço

O suporte à qualidade de serviço (QoS) das VANETs ainda é um desafio para os pesquisadores. É preciso desenvolver abordagens de roteamento adaptativo de QoS, que permutam rapidamente, criando novas rotas quando o caminho de roteamento atual torna-se

indisponível, como resultado de uma mudança na velocidade ou posicionamento do veículo, da topologia da rede ou da distância entre os veículos.

Outro tema que merece atenção é a definição de métricas de QoS para VANETs, dadas as grandes variações de métricas de desempenho sendo usadas pelos pesquisadores, incluindo as mais comuns como atraso e *jitter*[1].

Radiodifusão

Levando-se em consideração que um número expressivo de mensagens em VANETs é transmitido por radiodifusão, fica evidente a relevância desse tema e a motivação dos pesquisadores dessa área.

Para minimizar a quantidade de mensagens transmitidas por radiodifusão pela rede, são necessários algoritmos mais adequados e eficientes, que consigam lidar melhor com as colisões de mensagens.

Faz-se necessário, portanto, mais pesquisas para investigar esquemas inteligentes de inundações de pacotes e algoritmos distribuídos que consigam lidar eficientemente com esse tipo de comunicação.

Além disso, fornecer transmissão por radiodifusão confiável de mensagens com um mínimo de sobrecarga também gera muitos desafios técnicos, como, por exemplo, a escolha do veículo ao lado, a manutenção da comunicação entre os veículos que deixam e os que se juntam ao grupo, entre outros.

2.6 Considerações Finais

Este capítulo teve início com uma breve descrição das MANETs e, na sequência, uma explanação mais detalhada sobre as VANETs. Foram abordadas as principais

[1] O *jitter* é uma medida de variação do atraso, na entrega, entre os pacotes de dados enviados sucessivamente.

características das redes veiculares, seus três tipos de comunicação mais comuns e exemplos de aplicações, como aviso de local perigoso, aproximação de veículo de emergência e aviso de engarrafamento.

Com relação à segurança das VANETs, foram identificados alguns ataques realizados nas trocas de mensagens, como são os casos de informações falsas transmitidas pela rede, de informações erradas do sensor, de divulgação da identificação de outros veículos, da negação de serviço e do mascaramento.

Foram abordados, também, os principais desafios apontados pela literatura referentes à pesquisa sobre as VANETs, como, por exemplo, os protocolos de roteamento, os *frameworks* de segurança, a qualidade de serviço e a transmissão de mensagens por radiodifusão.

3 FERRAMENTAS DE SIMULAÇÃO

Neste capítulo é realizada uma explanação a respeito das simulações e dos tipos de ferramentas usados no estudo das VANETs. Na seção 3.1 são mostrados os objetivos e a importância dos simuladores de rede e de mobilidade. Na seção 3.2 é descrita a segunda versão do *Network Simulator*, ou simplesmente NS-2. Na seção 3.3 é descrito o simulador de mobilidade VanetMobiSim, mostrando as características dos modelos de macromobilidade e micromobilidade. Na seção 3.4 são apresentadas as considerações finais do capítulo.

3.1 Importância das Simulações

Para que uma dada tecnologia consiga atender as expectativas, uma série de experimentos deve ser realizada para testá-la. Dessa forma, as simulações de *software* desempenham um papel fundamental à medida que conseguem imitar com realismo e precisão diversos cenários, produzindo resultados semelhantes aos do mundo real (HASSAN, 2009).

No caso das VANETs, dois tipos de simulações são necessárias para se alcançar bons resultados:

- Simulação de tráfego (ou mobilidade): utilizada no controle e no planejamento do tráfego rodoviário;
- Simulação de rede: usada para avaliar os protocolos e aplicações de rede em condições variadas.

Essas simulações podem trabalhar de forma independente, no entanto, é necessária uma solução para utilizá-las juntas. Um simulador VANET tem essa característica, ou seja, um *framework* integrado capaz de unir as duas simulações mencionadas anteriormente.

O problema é que dificilmente um simulador desse tipo consegue atingir todos os requisitos necessários, sendo que a maioria deles tem problemas de interação entre os dois modelos de simuladores (HASSAN, 2009).

Martinez et al. (2009) fazem um estudo mais completo a esse respeito. O artigo mostra um comparativo entre alguns simuladores de VANETs disponíveis, além de comparar também alguns simuladores de rede e geradores de mobilidade.

Os autores ressaltam que até mesmo os resultados obtidos usando cada um desses simuladores podem ser diferentes, uma vez que eles foram desenvolvidos com propósitos distintos. Por exemplo, o TraNS (PIÓRKOWSKI et al., 2008) e o GrooveNet (GROOVENET, [2014?]) foram desenvolvidos para simulação de VANETs, já o NCTUns (NCTUNS 6.0, [2014?]) foi criado para simulações em redes de propósitos gerais e o MobiREAL (MOBIREAL, [2014?]) foi projetado para simular MANETs.

Concluindo essa questão, Zeadally et al. (2010) afirmam, taxativamente, que existem muitos simuladores, porém, nenhum deles fornece uma solução completa para o estudo das redes veiculares.

O simulador escolhido para a implementação do protocolo µTESLA foi o NS-2, pois, segundo Martinez et al. (2009), simuladores de redes (como o NS-2) são úteis para testar novos protocolos ou para propor modificações naqueles já existentes.

O simulador VanetMobiSim (HARRI; FIORE, 2005/06) foi usado para gerar os traços de mobilidade de tráfego. Esses traços de mobilidade são importantes à medida que geram cenários de simulação mais realistas, acrescentando semáforos, sinais de parada, limites de velocidade, filas de carros, congestionamento, entre outros.

Dessa forma, o arquivo gerado pelo VanetMobiSim foi usado como dados de entrada para o NS-2, que realizou a simulação de tráfego de rede e transferência de pacotes de dados usando o protocolo µTESLA.

3.2 *Network Simulator* Versão 2 (NS-2)

NS-2 é um simulador de rede. Este tipo de simulador permite aos pesquisadores estudarem o comportamento de uma rede em diferentes condições, de acordo com as configurações dos cenários de simulação.

De forma geral, os simuladores de rede são úteis para testar novos protocolos de rede e propostas de modificações nos protocolos já existentes de uma maneira controlada e reproduzível (MARTINEZ et al., 2009).

3.2.1 Origem do NS-2

De início, o NS (como NS-2 era chamado) era uma variante do simulador de rede REAL (KESHAV, 1997). Em 1995 o seu desenvolvimento foi apoiado pela DARPA (*Defense Advanced Research Projects Agency*) por meio do projeto VINT (*Virtual InterNetwork Testbed*) com uma colaboração entre LBNL (*Lawrence Berkeley National Laboratory*), Xerox PARC (um centro de pesquisas da Xerox), UCB (*University of California, at Berkeley*) e USC/ISI (*University of Southern California/Information Sciences Institute*).

Atualmente, seu desenvolvimento é mantido pela DARPA, SAMAN (*Simulation Augmented by Measurement and Analysis for Networks*), NSF (*National Science Foundation*), CONSER (*Collaborative Simulation for Education and Research*) e ACIRI (*AT&T Center for Internet Research at ICSI – International Computer Science Institute*).

O NS também incluiu contribuições de outros pesquisadores, como o código *wireless* da UCB *Daedelus* e projetos *Monarch* (*MObile Networking ARCHitectures*) e *Sun Microsystems* (THE NETWORK SIMULATOR, [2014?]).

3.2.2 Arquitetura

O simulador NS-2 é estruturado em dois módulos principais: um simulador escrito em linguagem C++ e um interpretador de OTcl. A necessidade de duas linguagens de programação se explica pela especificidade de cada uma.

A linguagem C++ é compilada e de uso tradicional, mostrando-se uma ferramenta bastante eficaz na programação dos módulos responsáveis pela execução dos protocolos ou aplicações. O comportamento detalhado de cada protocolo exige uma linguagem mais robusta e de baixo nível para ser usada na manipulação de *bytes*, processamento de pacotes e implementação de algoritmos que executam grande conjunto de dados.

Por outro lado, por ser uma linguagem interpretada, OTcl é mais adequada durante o processo de simulação, no qual ajustes de parâmetros são necessários com maior frequência. Isso simplifica e agiliza o trabalho do usuário, que não precisa recompilar seu modelo a cada mudança desse tipo (ALTMAN; JIMÉNEZ, 2003).

Em uma visão simplificada, o NS-2 é um interpretador de *script* Tcl (*Tool Command Language*) orientado a objeto (OTcl) que possui escalonador de eventos de simulação e bibliotecas de componentes de rede e de módulos de configuração de rede.

Dessa forma, para configurar e executar uma simulação de rede o usuário deve escrever um *script* em OTcl que inicia a execução de um escalonador de eventos, configura uma topologia de rede e informa a origem do tráfego.

Ao final da simulação, NS-2 produz um ou mais arquivos de saída em modo texto, contendo os detalhes da simulação. Estes dados podem ser usados para análise da simulação ou como entrada para uma ferramenta de exibição gráfica, chamada *Network Animator* (NAM). Esse processo é apresentado na Figura 9.



Figura 9. Visão simplificada do NS-2 (CHUNG; CLAYPOOL, [2014?]).

O usuário pode atuar em "tcl", escrevendo *scripts* e executando as simulações necessárias. O escalonador de eventos e a maioria dos componentes de rede são implementados em C++ e disponibilizados para OTcl através de uma ligação. Essa ligação, implementada usando TclCL (uma interface Tcl com C++), cria um relacionamento do objeto OTcl para cada objeto de C++. A arquitetura geral do NS-2 é apresentada na Figura 10.

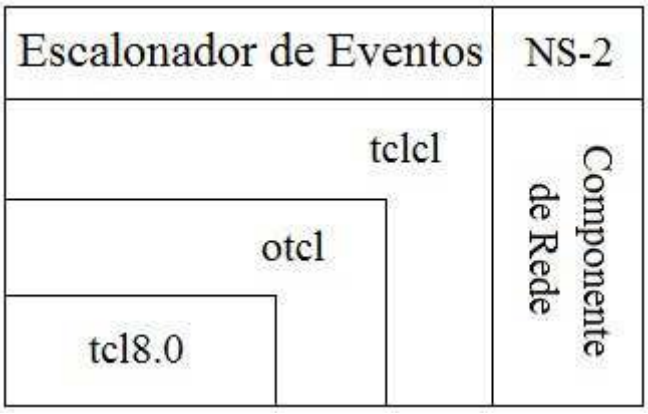


Figura 10. Arquitetura geral do NS-2 (CHUNG; CLAYPOOL, [2014?]).

NS-2, portanto, é um simulador de eventos discretos, destinado à pesquisa em rede, que fornece suporte para simulação de uma variedade de aplicações. O simulador implementa protocolos de rede e emula o comportamento de tráfego de aplicações, mecanismos de gerenciamento de fila, algoritmos de roteamento, entre outros (CHUNG; CLAYPOOL, [2014?]).

Apesar de todos os benefícios, segundo seus próprios desenvolvedores e responsáveis, o NS-2 não é um produto acabado, mas o resultado de um esforço contínuo de pesquisa e desenvolvimento.

3.3 VanetMobiSim

VanetMobiSim (*Vehicular Ad Hoc Networks Mobility Simulator*) é um gerador de traços de mobilidade, também chamado de simulador de tráfego. Segundo Martinez et al. (2009), esse tipo de simulador é necessário para aumentar o nível de realismo em simulações VANETs.

O VanetMobiSim é uma extensão do ambiente de simulação de mobilidade CanuMobiSim (CANU MOBILITY SIMULATION ENVIRONMENT, [2014?]), um *framework* flexível usado para modelagem de mobilidade, utilizado pelo grupo de pesquisa CANU (*Communication in Ad Hoc Networks for Ubiquitous Computing*) da Universidade de Stuttgart.

CanuMobiSim é baseado em Java e é capaz de gerar padrões de movimento[2] em diferentes formatos, fornecendo suporte às simulações de diferentes ferramentas para redes móveis, inclusive o NS-2.

[2] Movimentação programada dos nós da rede.

CanuMobiSim inclui analisadores de mapas em arquivos de dados geográficos (GDF – *Geographic Data Files*) e fornece implementações de vários modelos de mobilidade aleatórios, bem como modelos de movimentos dinâmicos – baseados em modelos de dinâmicas física e veicular – que definem padrões de mudança de velocidade e direção dos nós móveis.

Segundo Fiore et al. (2007), o foco da extensão VanetMobiSim é a mobilidade veicular, apresentando novos modelos realistas de movimento de automóveis, tanto no nível macroscópico quanto no microscópico.

No nível macroscópico, é possível importar mapas do banco de dados do *Census Bureau* dos EUA[3]. Além disso, ele fornece suporte para várias faixas de estradas, fluxos direcionais separados, restrições de velocidade diferenciadas e sinais de trânsito nos cruzamentos.

No nível microscópico, VanetMobiSim implementa novos modelos de mobilidade, fornecendo uma interação bastante realista de carro para carro e de carro para infraestrutura.

De acordo com esses modelos, os veículos regulam a sua velocidade dependendo dos carros nas proximidades, ultrapassam uns aos outros e agem de acordo com sinais de trânsito na presença de interseções (FIORE et al., 2007).

O IDM-LC (*Intelligent Driver Model with Lane Changes*), por exemplo, é um modelo de mobilidade em nível microscópico que permite a mudança de faixa e a ultrapassagem entre os veículos.

Ele inclui o gerenciamento dos cruzamentos regulados por sinais de trânsito (com sinais de parada ou semáforos) e de estradas com faixas múltiplas, representando a interação entre fluxos convergentes de veículos e atuando de forma consistente com a infraestrutura rodoviária.

---

[3] Agencia governamental encarregada pelo censo nos Estados Unidos.

3.3.1 Características de Macromobilidade

A macromobilidade trata da topologia da estrada, bem como da estrutura viária (estrada unidirecional ou bidirecional, faixas simples ou múltiplas), das características das estradas (limites de velocidade e restrições baseadas na classe dos veículos) e da presença de sinais de trânsito (sinais de parada e semáforos).

Além disso, o conceito de macromobilidade também inclui os efeitos da presença de pontos de interesse, que influenciam os padrões de movimento de veículos sobre a topologia da estrada (FIORE et al., 2007).

De acordo com Fiore et al. (2007), para se obter resultados realistas – e, portanto, mais interessantes – nas simulações de movimentos veiculares, é importante fazer uma boa seleção da topologia da estrada.

Isso é importante porque o comprimento das ruas, a frequência de intersecções, a quantidade de casas e prédios podem afetar algumas métricas de mobilidade, como a velocidade (mínima, máxima e média) dos automóveis, por exemplo.

Dessa forma, VanetMobiSim permite definir a topologia da estrada de quatro formas diferentes, sendo:

1. Grafo definido pelo usuário: a topologia é especificada listando os vértices e arestas do grafo (Figura 11a);

2. Mapa GDF: a topologia é importada de um arquivo específico para dados geográficos (Figura 11b);

3. Mapa TIGER: a topologia é extraída de um mapa obtido a partir do banco de dados TIGER, que contém descrições das áreas urbanas e rurais dos EUA (Figura 11c);

4. Grafo de Voronoi aglomerado: a topologia é gerada aleatoriamente pela criação de um mosaico de Voronoi[4] sobre um conjunto de pontos distribuídos não uniformemente (Figura 11d).

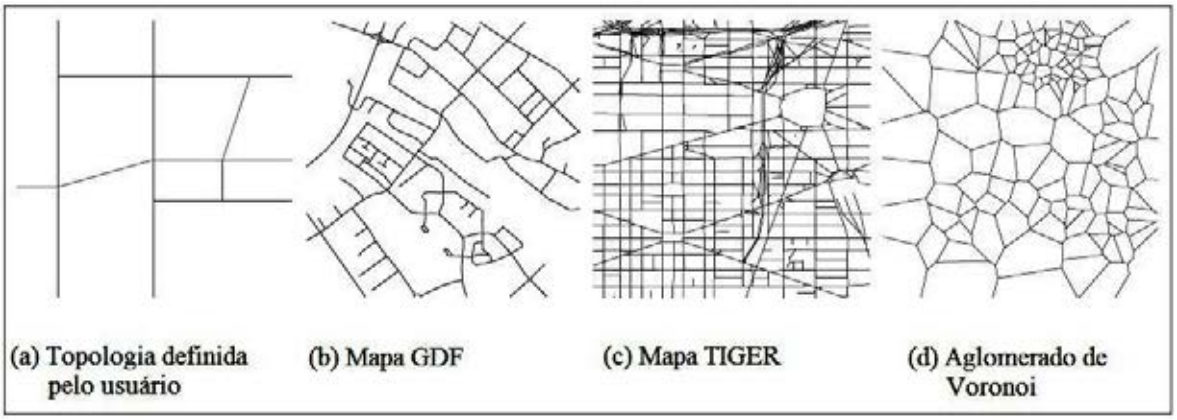


Figura 11. Topologias da estrada disponíveis no VanetMobiSim (FIORE et al., 2007, p. 3).

Com relação aos aspectos relacionados com a caracterização da estrutura da estrada, o simulador VanetMobiSim introduziu alguns aperfeiçoamentos com relação a CanuMobiSim, tais como introdução de pistas múltiplas, separação física dos fluxos de tráfego opostos em cada estrada, definição de limites de velocidade e implementação de sinais de trânsito em cada intersecção.

Nos cenários urbanos, os veículos tendem a se locomoverem entre pontos de interesse. VanetMobiSim explora a capacidade de CanuMobiSim de construir padrões de movimento, definindo alguns conjuntos de pontos de interesse, e módulos de cálculo de caminho, para calcular o melhor caminho entre tais pontos (FIORE et al., 2007).

[4] Mosaico de Voronoi é um tipo de decomposição do espaço, determinado pela distância entre os objetos nesse espaço.

3.3.2 Características de Micromobilidade

A micromobilidade veicular diz respeito aos aspectos relacionados com a modelagem da velocidade e aceleração dos carros, sendo responsável, também, pelas filas de carros, pelos engarrafamentos e pelas ultrapassagens.

Os modelos de micromobilidade podem ser divididos em três classes principais, segundo Fiore et al. (2007), de acordo com o aumento do grau de detalhe. Dessa forma, a velocidade individual dos veículos pode ser computada de um modo determinístico, como uma função de comportamento de veículos próximos em cenário de pista simples ou como uma função de comportamento de veículos próximos em um cenário de interação de fluxo múltiplo.

3.4 Considerações Finais

Neste capítulo foram apresentados os tipos de ferramentas utilizadas nas simulações das VANETs, a importância e os objetivos da simulação, bem como as dificuldades da escolha do simulador (ou simuladores) correto.

Também foram apresentadas as ferramentas de simulação utilizadas neste trabalho, isto é, o simulador de rede *Network Simulator* 2 e o simulador de tráfego VanetMobiSim, sendo apontadas as suas principais características.

4 PROTOCOLO µTESLA

Neste capítulo é apresentado o protocolo de autenticação de mensagens µTESLA. Na seção 4.1 é feita uma introdução à autenticação de mensagens, explicando a importância desse tipo de protocolo. Na seção 4.2 é explicado o protocolo TESLA e feita uma comparação entre este protocolo e o µTESLA, mostrando que o uso do µTESLA é mais vantajoso para as redes veiculares. Na seção 4.3 é descrito com mais detalhes o funcionamento do protocolo µTESLA. Na seção 4.4 são apresentadas as considerações finais do capítulo.

4.1 Autenticação de Mensagens

A autenticação de mensagens é um tema de fundamental importância no que tange a segurança de dados. De acordo com Stallings (2008) esse tema, juntamente com a assinatura digital, talvez seja a área mais confusa da segurança de rede, devido às complicações relacionadas aos diversos ataques e contramedidas.

O objetivo deste mecanismo é permitir que o receptor de uma mensagem saiba, com certeza, que a mensagem não foi alterada durante a transmissão e quem foi o emissor dessa mensagem, ou seja, o receptor sabe que os dados recebidos estão corretos e foram gerados pela fonte declarada.

De forma mais completa, segundo Stallings (2008, p. 226):

> A autenticação de mensagens é um mecanismo ou serviço usado para verificar a integridade de uma mensagem. A autenticação garante que os dados recebidos sejam exatamente iguais aos enviados (ou seja, não contêm modificação, inserção, exclusão ou repetição) e que a identidade afirmada pelo emissor é válida.

Quando mensagens são trocadas entre apenas duas partes, a autenticação dos dados pode ser alcançada com mecanismos de criptografia puramente simétricos, em que o emissor e o receptor compartilham uma chave secreta para calcular um código de autenticação de mensagem (MAC – *Message Authentication Code*) de todos os dados comunicados.

O MAC é uma técnica de autenticação que gera um pequeno bloco de dados de tamanho fixo com o uso de uma chave secreta. Assumindo que duas partes comunicantes (digamos *A* e *B*) compartilham uma chave secreta, quando *A* precisa enviar uma mensagem para *B* basta calcular o MAC como uma função da mensagem e da chave: *MAC = C(K, M)*, onde *MAC* é o código de autenticação de mensagem, *C* é uma função *MAC*, *K* é uma chave

secreta compartilhada e *M* é uma mensagem de entrada. A mensagem mais o MAC são enviados para *B*.

Quando recebe a mensagem, *B* realiza o mesmo cálculo sobre a mensagem recebida, usando a mesma chave secreta compartilhada, para gerar um novo MAC. O MAC recebido é comparado com o MAC calculado e caso sejam iguais o receptor (no caso, *B*) tem garantias de que a mensagem não foi alterada e partiu do emissor declarado (no caso, *A*). Esse processo pode ser observado na Figura 12.

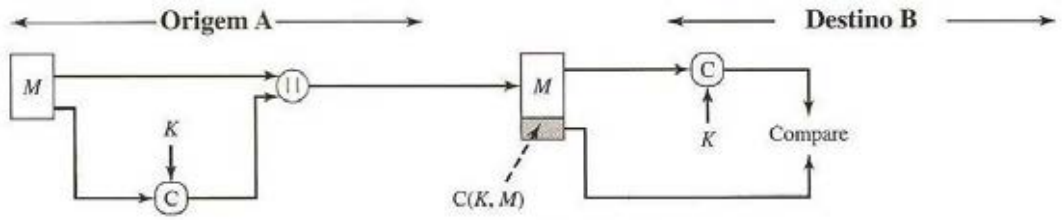


Figura 12. Autenticação da mensagem (STALLINGS, 2008, p. 232).

Contudo, este tipo de autenticação não pode ser utilizado em mensagens enviadas por radiodifusão, pois quando uma mensagem autenticada for enviada para vários receptores, qualquer um deles sabe qual é a chave MAC e pode representar outro remetente e forjar a origem das mensagens para outros receptores.

Por isso, é necessário um mecanismo assimétrico para se alcançar transmissão por radiodifusão autenticada (PERRIG et al., 2001). Os protocolos TESLA e μTESLA prometem fornecer autenticação de mensagens enviadas por radiodifusão utilizando mecanismos de criptografia simétricos. Esses protocolos são descritos nas próximas seções.

4.2 Legado do Protocolo TESLA

O TESLA (*Timed Efficient Stream Loss-tolerant Authentication*) é um protocolo de autenticação de comunicação por radiodifusão descrito em Perrig et al. (2000). De acordo com os autores, é um protocolo eficiente, tolerante a perda de pacotes e que pode ser escalado para um grande número de receptores. Apesar de utilizar funções de criptografia simétrica, também atinge propriedades assimétricas (PERRIG et al., 2000).

Entretanto, Perrig et al. (2001) assumem que o protocolo TESLA não foi projetado para ambientes computacionais limitados por três motivos:

1) TESLA autentica o pacote inicial com uma assinatura digital que exige um poder computacional muito grande em cada nó da rede;

2) O padrão TESLA revela uma chave por pacote, exigindo mais energia dos nós e gerando mais sobrecarga de processamento, de armazenamento e de rede;

3) A cadeia de chave unidirecional não cabe na memória de nós sensores simples, portanto TESLA puro seria impraticável à medida que esses nós não poderiam transmitir dados autenticados.

Focados nas questões de segurança, Perrig et al. (2001) desenvolveram um conjunto de protocolos de segurança para redes de sensores.

Esse projeto – denominado SPINS (*Security Protocols for Sensor Networks*) – tem duas estruturas de blocos seguras, SNEP (*Secure Network Encryption Protocol*) e µTESLA (uma versão adaptada do TESLA). O primeiro fornece importantes primitivas básicas de segurança, como confidencialidade e autenticação de dados entre duas partes, com baixa sobrecarga. O segundo é um protocolo que fornece transmissão por radiodifusão autenticada para ambientes com recursos limitados.

µTESLA foi projetado para resolver as deficiências do protocolo TESLA no que tange às redes de sensores. Nesse sentido, como resposta ao primeiro problema mencionado anteriormente, esse novo protocolo usa apenas mecanismos simétricos. Para solucionar o segundo problema, µTESLA revela a chave somente uma vez para cada intervalo. Além disso, restringe o número de remetentes autenticados para corrigir o terceiro problema (PERRIG et al., 2001).

4.3 Descrição do Protocolo µTESLA

Para uma melhor compreensão, o protocolo µTESLA será explicado por fases, sendo elas: configuração do emissor, inicialização de novos receptores, envio de pacotes e autenticação dos pacotes recebidos.

Configuração do emissor

O emissor deve gerar uma sequência de chaves secretas. Para gerar uma cadeia de chaves de tamanho *n*, o emissor escolhe a última chave K_n e gera os outros valores aplicando uma função unidirecional *F* sucessivamente, de forma que $K_j = F(K_{j+1})$. Dessa forma, é

possível calcular K_0, ... , K_j dado um K_{j+1}, mas não se pode calcular um K_{j+1} dado somente K_0, ... , K_j.

Inicialização de um novo receptor

Cada receptor deve ter uma chave autêntica da cadeia de chaves unidirecional como uma forma de compromisso para a cadeia inteira. Também, é necessário que o emissor e o receptor estejam vagamente sincronizados no tempo e que o receptor conheça o horário de divulgação da chave.

O receptor envia uma mensagem de solicitação de sincronização para o emissor contendo um NONCE[5], que responde com outra mensagem contendo sua hora atual T_s para sincronização de tempo, a chave K_i usada no intervalo passado i, a hora de início T_i do intervalo i, a duração T_{int} do intervalo de tempo e o atraso δ de divulgação da chave.

A sincronização de tempo é feita com o protocolo NTP (*Network Time Protocol*), que permite a sincronização dos relógios dos dispositivos de uma rede, como servidores, estações de trabalho e roteadores (NETWORK TIME PROTOCOL, [2014?]).

Na Figura 13, é mostrada uma troca de mensagens entre os veículos emissor e receptor para que seja realizada a sincronização.

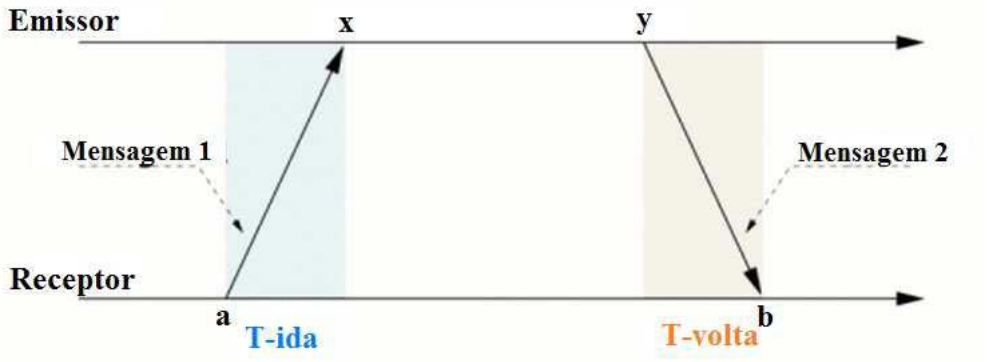


Figura 13. Troca de mensagens (NETWORK TIME PROTOCOL, [2014?]).

[5] NONCE é um número arbitrário (código) usado para assinar uma comunicação.

Primeiramente, o veículo receptor envia a ‘Mensagem 1’, contendo a sua hora atual ‘a’, para o veículo emissor. Este lê seu relógio, que fornece o instante ‘x’ e armazena ‘a’ e ‘x’ em variáveis. Após algum tempo o emissor lê seu relógio novamente, que fornece o instante ‘y’. O emissor, então, envia a ‘Mensagem 2’ para o receptor, contendo os valores de ‘a’, ‘x’ e ‘y’. Quando recebe a ‘Mensagem 2’, o receptor lê seu relógio, que fornece o instante ‘b’.

Mesmo tendo conhecimento desses quatro valores, não é possível calcular o tempo que ‘Mensagem 1’ e ‘Mensagem 2’ levaram para ser transmitidas (‘T-ida’ e ‘T-volta’, respectivamente). Porém, o tempo total de ida e volta (RTT – *Round Trip Time*) pode ser calculado da seguinte maneira:

$$rtt = (b-a) - (y-x)$$

Considerando que o tempo de ida é igual ao tempo de volta, pode-se calcular o deslocamento entre emissor e receptor como:

$$deslocamento = (x - a + y - b) / 2$$

O fato de o protocolo NTP considerar que ‘T-ida’ é igual a ‘T-volta’ implica em um erro, que no pior caso é igual à metade do rtt.

Envio de pacotes

O tempo é dividido em intervalos, sendo que o emissor associa cada chave da cadeia de chaves com um intervalo de tempo. Ou seja, no intervalo de tempo *t*, o emissor usa a chave K_t para calcular o MAC dos pacotes desse intervalo.

A chave K_t será revelada após um atraso de δ intervalos depois do final do intervalo de tempo *t*. O tempo de atraso δ de divulgação da chave é de alguns poucos intervalos de tempo, sendo maior do que qualquer tempo entre as trocas de mensagens de emissores e receptores.

Autenticação dos pacotes recebidos

Quando um receptor recebe um pacote, ele deve ter certeza de que este não foi alterado, o que pode acontecer se alguém já tiver conhecimento da chave divulgada de um intervalo de tempo.

Por isso, o receptor precisa estar certo de que o emissor não divulgou a chave correspondente ao pacote recebido, o que é chamado de condição de segurança. Se o pacote

recebido satisfaz a condição de segurança, o receptor armazena o pacote, caso contrário, o receptor descarta o pacote, pois o mesmo pode ter sido alterado.

Assim que o nó recebe uma chave K_j de um intervalo de tempo anterior, ele autentica a chave verificando se ela corresponde à última chave autêntica conhecida K_i, usando um pequeno número de aplicações da função unidirecional F: $K_i = F^{j-i}(K_j)$. Se a verificação for bem sucedida, a nova chave K_j é autêntica e o receptor pode autenticar todos os pacotes válidos que foram enviados dentro dos intervalos de tempo *i* para *j*. Ao final desse processo, o receptor também substitui a chave armazenada K_i por K_j.

4.3.1 Transmissão de Mensagens Autenticadas

µTESLA requer que os veículos emissor e receptor estejam ao menos fracamente sincronizados com relação ao tempo, sendo que cada veículo conhece o limite superior sobre o erro máximo de sincronização.

Para enviar um pacote com uma mensagem autenticada, o emissor calcula um MAC no pacote com uma chave que é secreta naquele momento. Quando um veículo recebe um pacote, pode verificar que a chave MAC correspondente não foi ainda revelada pelo emissor tendo como referência o seu relógio fracamente sincronizado, o seu erro máximo de sincronização e o horário em que as chaves foram divulgadas.

Uma vez que é assegurado a um veículo que a chave MAC é conhecida apenas pelo emissor, também lhe é assegurado que nenhum adversário poderia ter alterado o pacote em trânsito e este é armazenado em um *buffer*. No momento da divulgação da chave, o emissor transmite a chave de verificação para todos os veículos.

Quando um veículo recebe a chave divulgada, pode verificar a exatidão da chave realizando um cálculo com a função unidirecional *F*. Se a chave está correta, o veículo pode agora usá-la para autenticar o pacote armazenado em seu *buffer*.

Cada chave MAC pertence a uma cadeia de chaves, gerada por uma função pública unidirecional *F*. Para gerar a cadeia de chaves unidirecional, o emissor escolhe a última chave K_n da cadeia e aplica *F* repetidamente para calcular todas as outras chaves: $K_i = F(K_{i+1})$.

4.3.2 Exemplo de Funcionamento

Na Figura 14 é apresentado um conjunto de chaves relacionadas com os intervalos de tempo, onde a chave K_1 está relacionada com o intervalo de tempo 1, a chave K_2 com o tempo 2, K_3 com tempo 3 e K_4 com tempo 4.

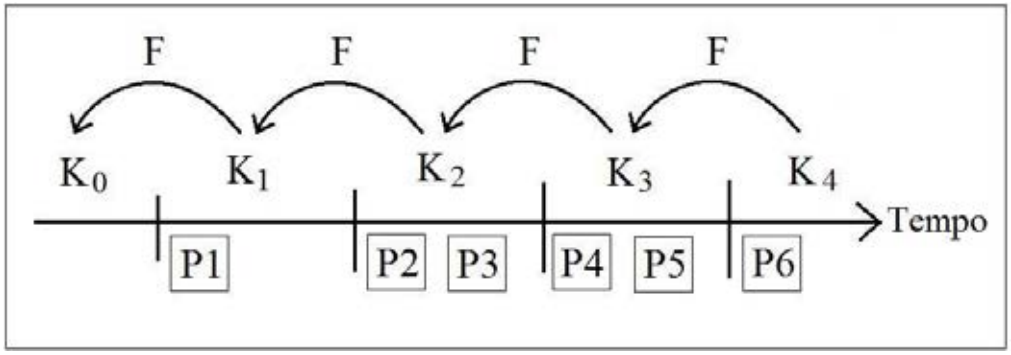


Figura 14. Exemplo de funcionamento do µTESLA.

A chave K_0 é a chave simétrica global – ou chave mestre – compartilhada por todos os participantes da rede. Todos os pacotes enviados no mesmo intervalo de tempo são autenticados com a mesma chave. Ou seja, o pacote *P1* foi enviado no tempo 1 e contém um valor MAC gerado com a chave K_1, os pacotes *P2* e *P3* foram enviados no tempo 2 contendo um MAC gerado com a chave K_2.

Da mesma forma, os pacotes *P4* e *P5* contêm um valor MAC gerado através da chave K_3 e o pacote *P6* contém um MAC gerado a partir da chave K_4. Além disso, *F* é uma função unidirecional utilizada, primeiramente, para gerar a cadeia de chaves e, na sequência, é usada para autenticar essas mesmas chaves após serem recebidas pelos veículos receptores.

Sabendo que neste exemplo as chaves são divulgadas a cada intervalo de 2 tempos, a chave K_1 deveria ser revelada no intervalo 3. Dessa forma, a chave seria autenticada calculando $K_0=F(K_1)$ e o pacote *P1* poderia, então, ser autenticado sem problemas.

O protocolo µTESLA ainda apresenta o seguinte mecanismo para evitar perda de pacotes: caso os pacotes do intervalo 3, por exemplo, tenham sido perdidos por problemas na rede, nenhum receptor, em princípio, poderia autenticar os pacotes enviados no tempo 1 (pacote *P1*, no caso), pois o pacote que revelaria a chave K_1 teria sido perdido.

Porém, no intervalo 4 a chave K_2 é revelada e pode ser autenticada usando a função unidirecional $K_0=F(F(K_2))$, podendo também recuperar K_1 fazendo $K_1=F(K_2)$. Dessa forma, os pacotes *P2* e *P3* poderiam ser autenticados com K_2 e o pacote *P1*, com K_1.

4.4 Considerações Finais

Neste capítulo foi abordado o tema da autenticação de mensagens, mostrando sua importância e seus objetivos no tocante à segurança de dados. Nesse contexto, foi explicado o mecanismo de autenticação denominado MAC, que é utilizado na comunicação entre duas partes compartilhando uma chave secreta.

O protocolo de autenticação μTESLA é uma versão adaptada do TESLA. Por essa razão, este capítulo discorreu, primeiramente, sobre os principais pontos referentes ao TESLA e na sequência descreveu detalhadamente o μTESLA, explicando suas fases de configuração do emissor de uma mensagem, de envio de pacotes, do processo de inicialização de um novo receptor e de autenticação dos pacotes recebidos.

5 DESENVOLVIMENTO E IMPLEMENTAÇÃO

O presente capítulo contém informações sobre a parte prática do projeto de mestrado. Na seção 5.1 é explicada a fase de implementação do protocolo µTESLA, ilustrada com os diagramas de classe e de sequência. Na seção 5.2 é descrita a especificação do protocolo, com a definição dos campos dos pacotes de dados. Na seção 5.3 é feita uma descrição do protocolo CMAP, exibindo os diagramas de classe. Na seção 5.4 são apresentadas as considerações finais do capítulo.

Para que a explicação fique mais clara, usaremos o termo ‘estação base’ para representar a entidade emissora de uma mensagem, enquanto a expressão ‘veículo’, usaremos para representar qualquer veículo que possa receber tal mensagem.

5.1 Detalhes da Implementação do Protocolo µTESLA

A implementação do protocolo µTESLA é uma parte fundamental do trabalho proposto. Para validar o modelo implementado existem três técnicas mais comuns:

1. Comparação com sistema real;
2. Comparação com outros resultados na literatura;
3. Comparação com resultado esperado.

Uma comparação entre o modelo implementado e os dados de um sistema real seria inviável, devido à complexidade e aos custos gerados pelo deslocamento de diversos veículos equipados e habilitados para construírem uma rede em uma rodovia ou um ambiente urbano.

Por outro lado, uma comparação com os resultados de outros trabalhos seria bastante interessante, contudo, nenhum trabalho dessa natureza foi encontrado na literatura, para que fosse possível realizar essa comparação.

Dessa forma, o modelo implementado do protocolo µTESLA foi comparado com o resultado esperado de tal protocolo, isto é, algumas situações foram criadas e o comportamento esperado para o protocolo foi comparado com os resultados obtidos pelas simulações. A tabela 1 relaciona as situações criadas e o comportamento esperado para o protocolo.

TABELA 1. Descrição do comportamento esperado para cada situação.

SITUAÇÃO	COMPORTAMENTO ESPERADO
Mensagem recebida pelo veículo sem a devida sincronização.	O veículo descarta a mensagem.
Mensagem recebida pelo veículo após a devida sincronização.	O veículo armazena a mensagem em seu *buffer*.
Chave autêntica recebida pelo veículo.	O veículo utiliza as mensagens autênticas e descarta as falsas.
Chave falsa recebida pelo veículo.	O veículo descarta a chave.

Para a implementação do protocolo μTESLA, foi utilizada a linguagem de programação C++, pois é a linguagem usada na construção do NS-2, juntamente com a linguagem OTcl.

Segundo Oliveira (2011), a linguagem C++ é adequada para a programação de sistemas de baixo nível, pois permite a manipulação de *bytes*, o processamento de pacotes e a implementação de algoritmos. Portanto, pode ser usada para alterar o comportamento de um módulo existente ou de outra operação com manipulação de pacotes dentro do NS-2, sendo a linguagem ideal para a construção de módulos responsáveis pela execução de novos protocolos.

Para a criação de um novo protocolo é necessária a criação de cabeçalho e agentes novos. Os cabeçalhos determinam a estrutura de dados utilizada por cada pacote e os agentes são terminais da camada de rede que utilizam (enviando e/ou recebendo) esses pacotes.

A seguir são apresentados os diagramas de classe dos arquivos criados para o protocolo μTESLA.

Na Figura 15 pode ser observado o diagrama de classe de MTeslaHeaderClass. Esta classe tem a função de definir o cabeçalho dos pacotes que são enviados utilizando o protocolo μTESLA.

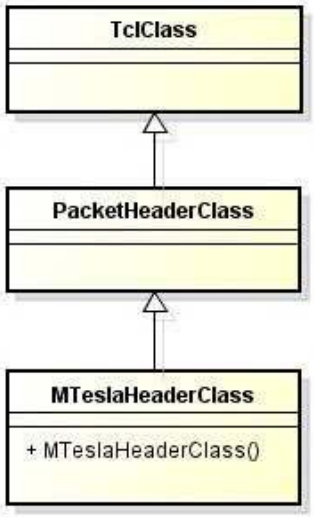


Figura 15. Diagrama de classe de MTeslaHeaderClass.

As classes Estacao_BaseClass e VeiculoClass criam *links* (ligações) entre as variáveis usadas nas diferentes linguagens Tcl e C++, podendo ser visualizadas no diagrama de classe da Figura 16. Cada classe possui seu método construtor e outro denominado ‘create’ que retornam uma referência de um objeto ‘TclObject’.

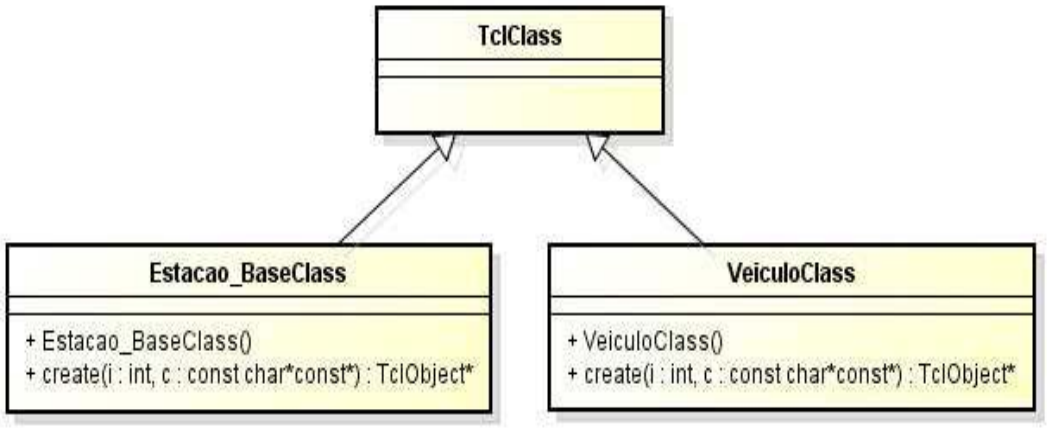


Figura 16. Diagrama de classe de Estacao_BaseClass e VeiculoClass.

Na Figura 17 é apresentado o diagrama de classe de Estacao_Base e de Veiculo. Essas classes são necessárias para criação de novos agentes, possuindo métodos de acesso ao cabeçalho e métodos específicos do protocolo.

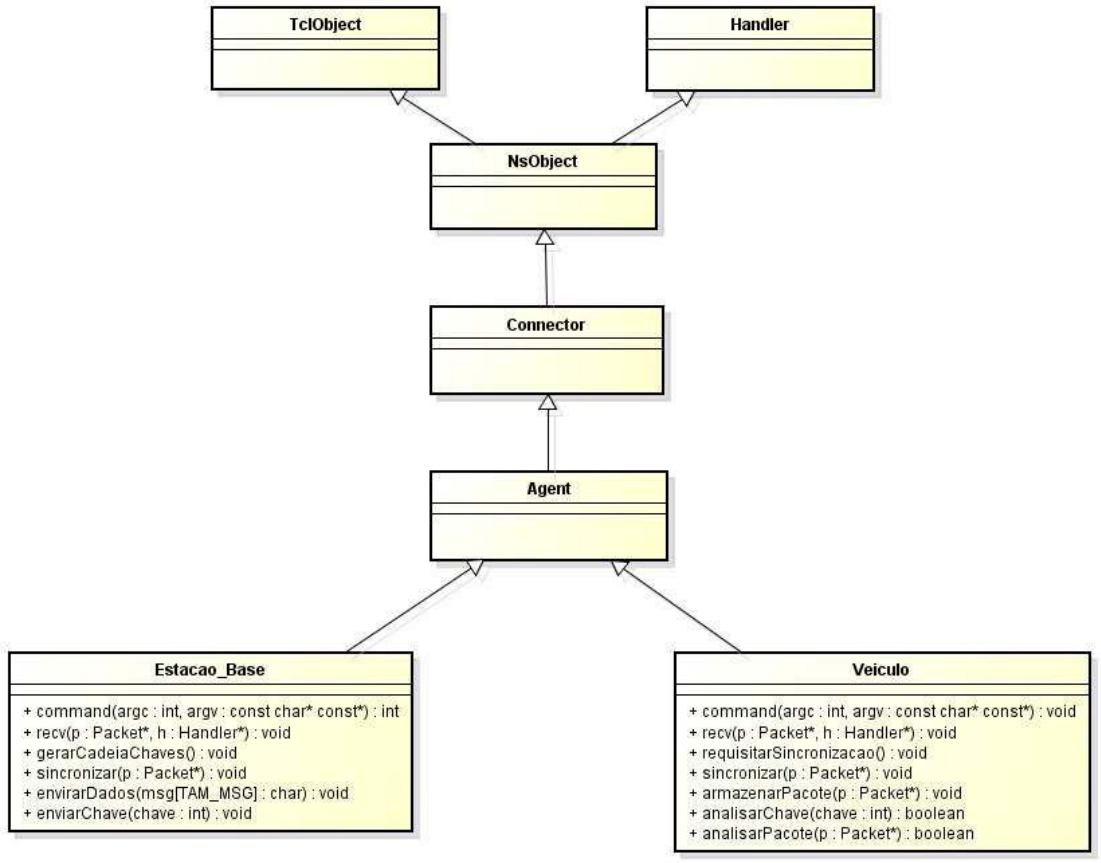


Figura 17. Diagrama de classe de Estacao_Base e de Veiculo.

As duas classes possuem os métodos básicos ‘command’ e ‘recv’, sendo que o primeiro trata dos comandos gerados em OTcl (não compilados) via *script* de simulação e o segundo trata dos comandos feitos em C++ (compilados).

Além desses dois métodos indispensáveis para a interação com o simulador, as classes Estacao_Base e Veiculo também possuem outros métodos, que constituem o protocolo μTESLA.

Os principais métodos da classe Estacao_Base são:

- gerarCadeiaChaves: a estação base gera a cadeia de chaves que será usada para autenticar os pacotes de dados;
- sincronizar: a estação base responde a requisição de sincronização feita pelo veículo;
- enviarDados: a estação base envia os pacotes de dados contendo mensagens de texto para o veículo;

- enviarChave: a estação base envia uma chave de autenticação pertencente à cadeia de chaves.

Os principais métodos da classe Veiculo são:

- requisitarSincronizacao: o veículo envia um NONCE e requisita sincronização com a estação base;
- sincronizar: o veículo recebe a resposta da estação base e finaliza a sincronização;
- armazenarPacote: o veículo armazena em seu *buffer* o pacote de dados que recebeu para autenticá-lo depois;
- analisarChave: o veículo analisa a chave de autenticação recebida;
- analisarPacote: o veículo analisa os pacotes de dados armazenados com a chave recebida.

As cinco classes mencionadas anteriormente são os arquivos do protocolo μTESLA, necessários para o seu funcionamento em conjunto com o simulador. Além disso, outras mudanças são necessárias em alguns arquivos preexistentes no simulador, a saber, os arquivos packet.h, tcl/lib/ns-default.tcl, tcl/lib/ns-packet.tcl e Makefile.

No início do arquivo packet.h deve-se acrescentar:

```
#define HDR_MTESLA(p) (hdr_mTesla::access(p))
static const packet_t PT_MTESLA = 73;
```

No mesmo arquivo, deve-se acrescentar no método initName() da classe p_info{}:

```
name_[PT_MTESLA} = “mTesla”;
```

No arquivo ns-default.tcl, adicionam-se as linhas:

```
Agent/Estacao_Base set packetSize_ 128;
Agent/Veiculo set packetSize_ 128;
```

Em ns-packet.tcl, adiciona-se em protolist{}:

```
MTesla
```

Por fim, no arquivo Makefile, deve-se acrescentar a linha:

```
mTesla/estacaoBase.o mTesla/veiculo.o \
```

Após essas mudanças, é preciso recompilar o simulador para que ele as reconheça e possa funcionar corretamente.

Na Figura 18 é mostrado um diagrama de sequência com um fluxo simples de execução de trocas de mensagens usando o protocolo μTESLA.

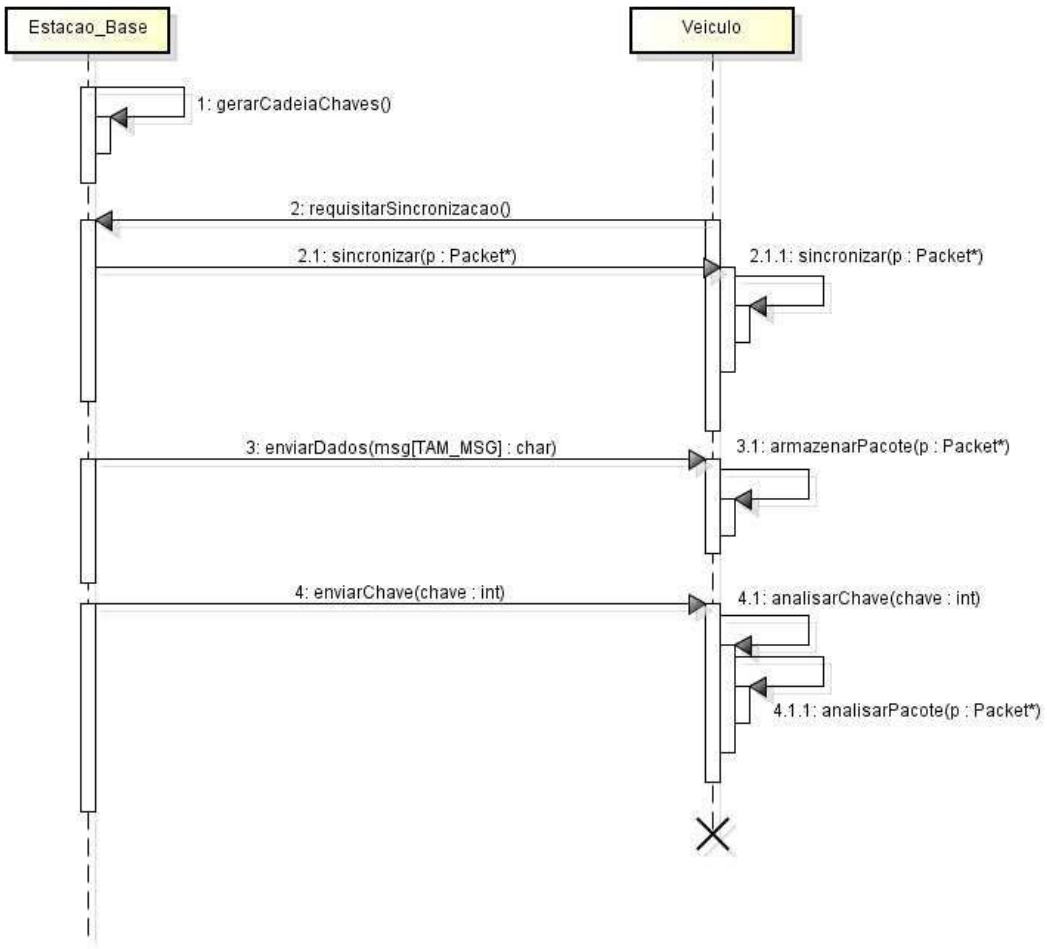


Figura 18. Diagrama de sequência das trocas de mensagens.

A classe ‘Estação Base’ representa as unidades de beira de estrada e a classe ‘Veiculo’ representa os veículos participantes da rede. Dessa forma, a estação base inicia o processo gerando uma cadeia de chaves. Estas chaves serão usadas mais tarde para fazer a autenticação dos pacotes de dados recebidos e armazenados pelos veículos.

Em um dado momento, um veículo qualquer entra na rede e requer uma sincronização com a estação base que responde concluindo a sincronização. A partir desse momento o veículo está apto a receber mensagens da estação base.

A estação base pode enviar diversos pacotes de dados contendo mensagens para os veículos e pode, também, enviar pacotes de dados contendo as chaves específicas para as autenticações das mensagens.

O veículo, por sua vez, quando recebe um pacote de dados o armazena em seu *buffer* para uma posterior verificação de autenticidade. Quando recebe um pacote contendo uma chave de autenticação, o veículo verifica se a chave é autêntica.

Caso o resultado seja positivo, o veículo busca no *buffer* os pacotes de dados referentes a essa chave, autenticando os pacotes verdadeiros e descartando os pacotes falsos; caso contrário, o veículo simplesmente descarta esse pacote contendo a chave.

5.2 Especificação do Protocolo µTESLA

Para a implementação do protocolo µTESLA foi necessário definir os tipos de pacotes de dados usados pelo protocolo, visto que esse tipo de informação não foi fornecido pelos criadores do protocolo.

A seguir encontra-se a definição desses pacotes, sendo que a primeira linha corresponde aos nomes dos respectivos campos e a segunda, aos tamanhos.

Definição dos pacotes de dados

A primeira etapa a ser cumprida em uma comunicação usando o protocolo µTESLA é a sincronização. Estação base e veículo precisam trocar informações a respeito de seus relógios para ficarem, ao menos, fracamente sincronizados com relação ao tempo.

Sendo assim, quando entrar na rede, o veículo envia um pacote de dados com um NONCE para a estação base solicitando sincronização. Esse pacote contém 3 campos, conforme ilustrado na Figura 19.

tipo	ID	NONCE
1 *byte*	16 *bytes*	20 *bytes*

Figura 19. Pacote de dados com requisição de sincronização.

O primeiro campo é o identificador do tipo do pacote de dados, com tamanho de 1 *byte*. Por meio dessa informação, o veículo consegue distinguir os tipos diferentes de pacotes e seus conteúdos. Esse campo deve existir em todos os pacotes exercendo a mesma função, sendo que a repetição dessa explicação nos pacotes subsequentes é, então, desnecessária.

O segundo campo é o ID, isto é, a identificação do veículo, com um tamanho de 16 *bytes*. Sendo do tipo UUID (*Universally Unique Identifier*) esse identificador é único para cada veículo.

O terceiro campo é o NONCE com 20 *bytes* de tamanho. O NONCE é um número aleatório que vai servir como uma espécie de desafio, pois a mensagem recebida por esse veículo logo em seguida deverá conter esse mesmo valor, caso contrário, o veículo não a aceitará.

Ao receber esse pacote de dados do veículo, a estação base responde enviando um pacote contendo 8 campos, como pode ser observado na Figura 20.

tipo	ID	hora_atual	chave_int_ant	hora_int_ant	duração_int	atraso_div_chave	MAC
1 *byte*	16 *bytes*	4 *bytes*	16 *bytes*	4 *bytes*	4 *bytes*	2 *bytes*	20 *bytes*

Figura 20. Pacote de dados para sincronização.

O segundo campo é o ID da estação base, seu identificador único contendo 16 *bytes* de tamanho.

O terceiro campo contém a hora atual do sistema da estação base, com 4 *bytes*. Essa informação é necessária para a sincronização.

O quarto campo contém a chave usada no intervalo anterior, com 16 *bytes*. O veículo deve armazená-la. Quando, tempos depois, receber outra chave para autenticar as mensagens, o veículo irá aplicar uma função unidirecional com a segunda chave para comparar o resultado com a primeira (recebida anteriormente).

Essa operação é do tipo $K_j = F(K_{j+1})$, onde K_j é a chave recebida no intervalo de tempo j, K_{j+1} é a chave recebida no intervalo seguinte e F é a função unidirecional. Caso os valores coincidam, a chave é autêntica, pois pertence à cadeia de chaves gerada pela estação base. Caso contrário, a chave é falsa e deve ser descartada.

No quinto campo encontra-se a hora do início do intervalo anterior, tendo 4 *bytes* de tamanho. No sexto campo tem-se a duração de cada intervalo de tempo, com um tamanho de 4 *bytes*. O atraso de divulgação das chaves, com 2 *bytes* de tamanho, encontra-se no sétimo campo. As informações desses três últimos campos são necessárias para que o veículo possa criar seu cronograma de divulgação das chaves de autenticação.

Finalmente, no último campo tem-se um valor MAC, com 20 *bytes* de tamanho. Esse valor MAC é o resultado da concatenação de todos os itens dos campos mencionados anteriormente mais o NONCE enviado pelo veículo na mensagem anterior (Figura 19).

Após receber esse pacote de dados, o veículo calcula o valor MAC dos 6 primeiros campos desse pacote concatenado com o NONCE que ele mesmo gerou anteriormente e compara com o MAC recebido. Se os valores coincidirem é finalizada a sincronização e o veículo está apto a receber mensagens da estação base para futura autenticação. Caso contrário, a comunicação é cancelada.

Quando a estação base deseja mandar uma mensagem para o veículo, ela envia um pacote de dados contendo 3 campos, conforme Figura 21.

tipo	mensagem	MAC
1 *byte*	100 *bytes*	20 *bytes*

Figura 21. Pacote de dados contendo mensagem.

No segundo campo tem-se a mensagem propriamente dita, com um tamanho de 100 *bytes* (tamanho suficiente para armazenar uma mensagem de alerta) e no terceiro campo, com 20 *bytes* de tamanho, tem-se o valor MAC que é o resumo da mensagem do primeiro campo, utilizando a chave do intervalo correspondente.

No momento marcado para a revelação da chave de um determinado intervalo de tempo, a estação base envia um pacote de dados, com 2 campos, conforme ilustrado na Figura 22.

tipo	chave
1 *byte*	16 *bytes*

Figura 22. Pacote de dados contendo chave.

O segundo campo contém uma chave de 16 *bytes* de tamanho. O veículo vai receber esse pacote e usar a chave para autenticar as mensagens que foram recebidas e armazenadas anteriormente em seu *buffer*.

Diagrama de estados

Utilizando-se do protocolo μTESLA, veículo e estação base se comunicam trocando mensagens, cujos pacotes de dados foram especificados anteriormente. De forma geral, essa comunicação é feita de acordo com o diagrama de estado ilustrado na Figura 23.

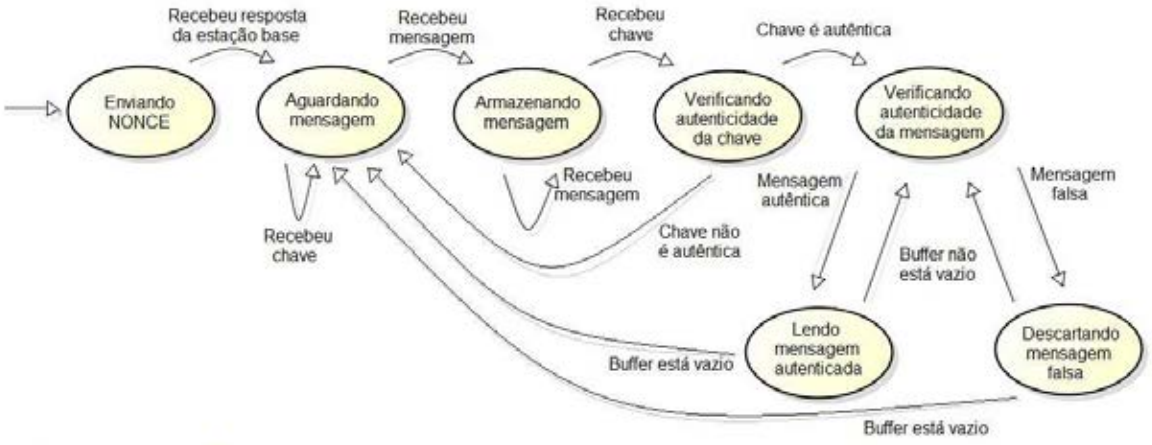


Figura 23. Diagrama de estado do protocolo μTESLA.

No primeiro estado (‘Enviando NONCE’) o veículo está conectado a uma rede e começa a enviar mensagens contendo um NONCE até receber uma resposta de uma estação base. Quando isso acontece, o veículo passa para o estado ‘Aguardando mensagem’, no qual permanece à espera de novas mensagens.

Caso receba uma chave de autenticação, o veículo permanece nesse estado. Caso receba uma mensagem, o veículo passa para o estado ‘Armazenando mensagem’, em que o veículo armazena, em seu *buffer*, a mensagem recebida e permanece nesse estado enquanto estiver recebendo novas mensagens.

Quando recebe uma chave de autenticação, o veículo passa para o próximo estado (‘Verificando autenticidade da chave’) para verificar se a chave recebida é autêntica. Em caso positivo, o veículo passa para o estado ‘Verificando autenticidade da mensagem’; em caso negativo, o veículo volta para o estado ‘Aguardando mensagem’.

No estado ‘Verificando autenticidade da mensagem’, o veículo verifica todas as mensagens armazenadas em seu *buffer* com a chave de autenticação. Caso a mensagem seja autêntica, o veículo passa para o estado ‘Lendo mensagem autenticada’ e utiliza essa mensagem. Caso contrário, o veículo passa para o estado ‘Descartando mensagem falsa’ e a mensagem falsa é apagada do *buffer*.

Estando o veículo no estado ‘Lendo mensagem autenticada’ ou no estado ‘Descartando mensagem falsa’, se o *buffer* não estiver vazio, o veículo volta para o estado ‘Verificando autenticidade da mensagem’ para verificar as mensagens restantes; se não

houver mais mensagens no *buffer*, o veículo volta para o estado ‘Aguardando mensagem’ para receber novas mensagens.

Como pôde ser observado na Figura 23, não foi criado o estado final do diagrama de estados, pois geraria diversas transições tornando o diagrama incompreensível. Contudo, o estado final seria o momento em que o veículo perde a conexão com a rede, o que pode acontecer em qualquer estado. Nesse momento, o veículo volta novamente para o primeiro estado (‘Enviando NONCE’), onde permanece até uma nova conexão.

5.3 Descrição do Protocolo CMAP

Os resultados do protocolo μTESLA foram comparados com os resultados do protocolo CMAP (HAO et al., 2011). A seguir é explicado o funcionamento do protocolo CMAP.

Duas etapas precisam ser cumpridas antes das comunicações terem início. Primeiro, a Autoridade Certificadora dá início ao sistema criptográfico gerando as chaves do sistema e publicando as chaves públicas para as estações bases, que ficam responsáveis pelo gerenciamento das chaves.

Segundo, quando um usuário deseja se registrar em um grupo, a estação base gera a chave privada do grupo e a transmite para o usuário. De posse dessa chave, o usuário está apto a assinar mensagens regulares.

Autenticação de mensagem cooperativa

O CMAP é um protocolo de autenticação de mensagem de modo cooperativo. Foi proposto para ser usado nas redes veiculares em cenários de estrada (HAO et al., 2011) e de cidade (HAO; HAN; CHENG, 2012), pressupondo que cada mensagem de segurança carrega a informação de localização do veículo.

Dessa forma, cada veículo envia uma mensagem regular (RBM – *Regular Broadcast Message*) a cada 300 milissegundos, contendo sua informação geográfica atual. Quando seus vizinhos recebem uma RBM, eles decidem se são verificadores dessa mensagem através de um algoritmo de seleção de verificadores. Esses verificadores são definidos de acordo com suas localizações em relação ao veículo emissor. Somente os verificadores selecionados

verificam a validade da mensagem, enquanto que os não-verificadores dependem dos resultados desses verificadores.

Se um veículo é o verificador da mensagem, ele pode autenticá-la imediatamente, caso seja uma mensagem válida; ou pode descartá-la e enviar mensagens de autenticação cooperativas (CAM – *Cooperative Authentication Message*) para os outros veículos, caso a mensagem seja falsa.

Os veículos não-verificadores esperarão pelas mensagens de alerta transmitidas pelos verificadores. Esses veículos só poderão usar a mensagem caso não recebam nenhuma CAM dentro de 100 milissegundos.

Acusação

Quando um veículo verifica que outro está transmitindo mensagens falsas maliciosamente, ele tem a função de reportar essa situação a uma Autoridade Certificadora.

Nesse processo, primeiramente o veículo reporta o fato à Autoridade Certificadora, que informa o ocorrido à estação base. Esta, por sua vez, envia chaves para o veículo acusado e o veículo acusador e envia as informações de volta à Autoridade Certificadora. Esta, por fim, calcula o *hash* das chaves e verifica qual é o veículo malicioso de fato.

Implementação

Foram criadas sete classes para a implementação do protocolo CMAP no NS-2. A classe CMAPHeaderClass é responsável pela definição do cabeçalho dos pacotes de dados transmitidos pelos nós da rede. Essa classe deriva da classe PacketHeaderClass, que, por sua vez, deriva de TclClass, conforme ilustrado na Figura 24.

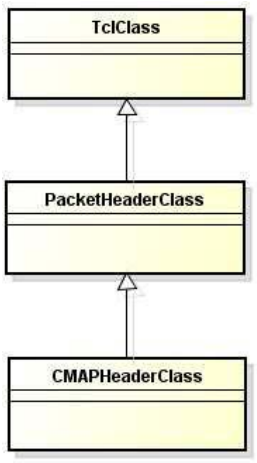


Figura 24. Diagrama de classe de CMAPHeaderClass.

As classes AutoridadeClass, RSUClass e VeiculoClass, representadas na Figura 25, criam os *links* necessários entre as variáveis usadas nas linguagens Tcl e C++. Essas três classes derivam da classe TclClass.

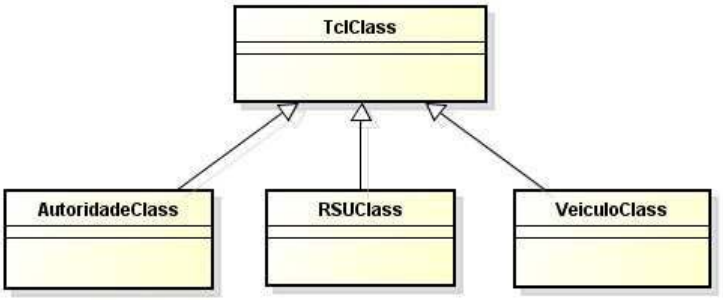


Figura 25. Diagrama de classe de AutoridadeClass, RSUClass e VeiculoClass.

Por fim, as classes Autoridade, RSU e Veiculo são os agentes usados nas simulações. Para isso, essas classes devem derivar da classe Agent, conforme ilustrado na Figura 26.

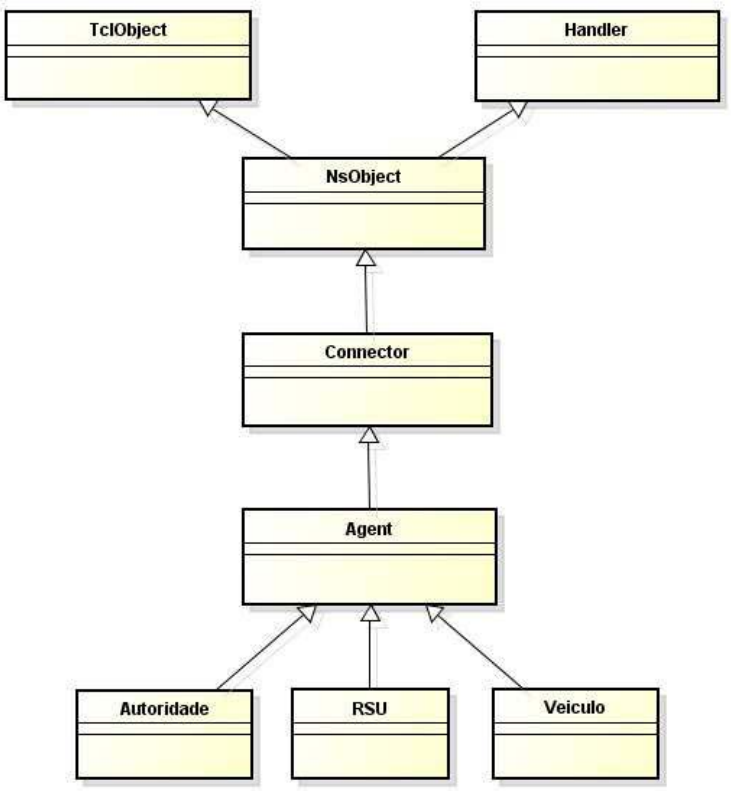


Figura 26. Diagrama de classe de Autoridade, RSU e Veiculo.

Com essas sete classes implementadas, iniciaram-se as simulações e comparações entre os protocolos CMAP e μTESLA.

5.4 Considerações Finais

Neste capítulo foram apresentados os detalhes de desenvolvimento e implementação desse trabalho. O protocolo μTESLA foi implementado em linguagem C++, pois é uma das linguagens usadas na construção do NS-2. Outra opção seria a linguagem OTcl, porém C++ é mais adequada para programação de sistemas de baixo nível, com operações de manipulação de *bytes*, de pacotes e de cabeçalhos.

Foram criadas as classes MTeslaHeaderClass, para definição do cabeçalho dos pacotes de dados, Estacao_BaseClass e VeiculoClass, para criação das ligações entre variáveis usadas nas duas diferentes linguagens e Estacao_Base e Veiculo, para a criação dos novos agentes.

Foi, também, especificado o protocolo μTESLA com as definições dos tipos diferentes de pacotes utilizados na transmissão de dados, sendo um tipo para requisição de sincronização, um para resposta da requisição e efetivação da sincronização, um para transmissão de mensagem e outro para transmissão da chave de autenticação.

Por fim, foi feita uma descrição do CMAP, um protocolo de autenticação de mensagem cooperativa, que também foi implementado no NS-2 para que fosse possível uma comparação entre os protocolos CMAP e μTESLA, com uma análise dos resultados obtidos por meio de simulações desses dois protocolos.

6 RESULTADOS

Neste capítulo mostram-se os resultados obtidos por meio das simulações no NS-2. Na seção 6.1 são descritas e explicadas as configurações usadas nas simulações. Na seção 6.2 é discutida a viabilidade do protocolo μTESLA, mostrando um teste de desempenho. Na seção 6.3 são comparados os protocolos μTESLA e CMAP por meio de testes de eficiência e de escalabilidade. Na seção 6.4 são apresentadas as considerações finais do capítulo.

6.1 Configurações do Cenário de Simulação

Todas as simulações foram realizadas com dois simuladores trabalhando em conjunto. Primeiramente, foi utilizado o simulador de tráfego VanetMobiSim, para fornecer o padrão de mobilidade dos veículos e, na sequência, foi utilizado o simulador de rede NS-2, para simular e gerar os resultados dos protocolos implementados, quais sejam, μTESLA e CMAP.

6.1.1 Simulação no VanetMobiSim

Nas simulações deste trabalho foi utilizado um modelo de mobilidade fornecido pelo VanetMobiSim, denominado Modelo de Direção Inteligente com Mudança de Faixa (IDM_LC – *Intelligent Driver Model with Lane Changing*).

Esse modelo fornece um cenário muito próximo ao encontrado no mundo real, pois permite a criação e a gestão de estradas com várias faixas e com cruzamentos regulados por sinais de trânsito, nas quais os veículos podem mudar de faixa e ultrapassar uns aos outros.

O modelo IDM_LC representa, ainda, a interação de diversos fluxos convergentes, nos quais os veículos trafegam de forma consistente com as normas de trânsito e da infraestrutura rodoviária, sempre calculando a velocidade levando em consideração os veículos próximos.

Quando um veículo está se dirigindo em direção a um cruzamento com semáforo, ele é informado sobre o estado do semáforo. Se a cor for verde, a passagem é concedida e a

velocidade atual é mantida; se a cor for vermelha, a passagem é negada e o carro desacelera até parar completamente no cruzamento.

Da mesma forma, quando vai passar por um cruzamento com sinal de parada, o veículo mantém sua velocidade e passa pelo cruzamento caso a preferência seja sua, ou diminui a velocidade até parar no cruzamento em caso contrário.

Configuração para as simulações

O cenário geral para as simulações foi configurado com uma área de 1000 x 1000 metros, faixas duplas nos dois sentidos e quatro cruzamentos com semáforos, sendo que os veículos trafegaram a uma velocidade mínima de 12 km/h e uma velocidade máxima de 50 km/h.

O tempo e o número de veículos das simulações variaram, dependendo do propósito de cada simulação. Por exemplo, para testar a viabilidade do protocolo μTESLA (seção 6.2) o tempo de simulação foi pequeno e para medir a escalabilidade dos protocolos μTESLA e CMAP (seção 6.3, item B) foram realizadas simulações com diferentes quantidades de veículos.

6.1.2 Simulação no NS-2

NS-2 permite a simulação de redes sem fio através de uma extensão criada pelo grupo Monarch (MONARCH PROJECT, [2014?]). Esta extensão criou um modelo de mobilidade que permite a simulação de nós móveis conectados por meio de interfaces de rede sem fio, incluindo simulações de redes *ad hoc* sem fio de múltiplos saltos.

Além disso, também é possível criar uma rede mista, onde uma rede sem fio é conectada a uma rede cabeada por meio de um nó especial, a estação base. É possível, ainda, configurar diversos parâmetros, como tipo de antena, modelo de propagação, protocolo de roteamento, entre outros.

As simulações utilizadas neste trabalho foram feitas em redes *ad hoc*, portanto, segue uma breve explicação dos parâmetros utilizados.

A primeira coisa a ser feita é a configuração dos nós móveis. Para isso, é necessário dar valor às variáveis, conforme é mostrado na Figura 27. O comando ‘set val’ é utilizado

para este propósito, sendo que a variável entre parênteses recebe a informação encontrada logo a seguir.

```
set val(chan)       Channel/WirelessChannel
set val(prop)       Propagation/Nakagami
set val(netif)      Phy/WirelessPhyExt
set val(mac)        Mac/802_11Ext
set val(ifq)        Queue/DropTail/PriQueue
set val(ll)         LL
set val(ant)        Antenna/OmniAntenna
set val(ifqlen)     50
set val(nn)         2
set val(rp)         DSDV
set val(x)          1000
set val(y)          1000
```

Figura 27. Configuração de variáveis.

Dessa forma, na primeira linha a variável ‘chan’ é configurada com um canal sem fio; na segunda linha a variável ‘prop’ é configurada com um modelo de propagação denominado Nakagami.

Na sequência, as variáveis ‘netif’ e ‘mac’ são configuradas com uma extensão do padrão IEEE 802.11; ‘ifq’ é uma fila de prioridade; ‘ll’ é uma camada de link; ‘ant’ é uma antena cujo sinal se propaga para todas as direções; ‘ifqlen’ é o tamanho da fila de pacotes; ‘nn’ é o número de nós móveis; ‘rp’ é um protocolo de roteamento do tipo DSDV e, finalmente, ‘x’ e ‘y’ definem a área de simulação.

Em seguida, pode-se definir uma topologia para a simulação, conforme o trecho de código mostrado na Figura 28. Na primeira linha é criada a topologia desejada e seus limites – val(x) e val(y) – usados na simulação são definidos na segunda linha.

```
set topo      [new Topography]
$topo load_flatgrid $val(x) $val(y)
```

Figura 28. Definição da topologia.

Na sequência, é criado um objeto denominado GOD (*General Operations Director*) que armazena as informações do ambiente de simulação. É passado como argumento o

número de nós móveis definido anteriormente na variável ‘nn’. A Figura 29 mostra a sintaxe para essa criação.

```
create-god $val(nn)
```

Figura 29. Criação do objeto GOD.

Uma vez que as variáveis já possuem os valores adequados, é preciso configurar os nós com esses valores. Na Figura 30 é mostrado como esse processo é realizado.

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

Figura 30. Configuração dos nós móveis.

A primeira linha define o protocolo de encaminhamento, a segunda, define o tipo da camada de enlace, seguida pelo protocolo de acesso ao meio físico, tipo da fila de espera, tamanho da fila de espera, tipo da antena, modelo de propagação, tipo de interface física e topologia criada.

Na sequência, é indicado o estado dos *traces*, ou seja, ativação/desativação da capacidade de registro das ações correspondentes: *agentTrace* (agentes definidos na simulação), *routerTrace* (nível de encaminhamento), *macTrace* (subnível MAC), *movementTrace* (movimento dos nós) e *phyTrace* (subnível físico).

Finalmente, em *channel* configura-se o tipo do canal.

Na Figura 31 é ilustrado um laço ‘for’ realizando a instanciação dos diferentes nós e inicializando as suas posições.

```
for {set i 0} {$i < $val(nn)} {incr i} {
        $ns_ initial_node_pos $node_($i) 50
}
```

Figura 31. Instanciação dos nós móveis.

Na Figura 32 são exibidos dois trechos de código, definindo nas três primeiras linhas a posição inicial (coordenadas X, Y e Z) do ‘nó 0’ e a posição inicial do ‘nó 1’ nas três últimas linhas.

```
$node_(0) set X_ 50.0
$node_(0) set Y_ 50.0
$node_(0) set Z_ 0.0

$node_(1) set X_ 55.0
$node_(1) set Y_ 50.0
$node_(1) set Z_ 0.0
```

Figura 32. Definição de posição dos nós móveis.

Estando os nós móveis configurados adequadamente, é possível criar os agentes e ligá-los aos nós. Na Figura 33 é exibido um trecho de código criando agentes ‘Estacao_Base’ e ‘Veiculo’ na primeira e na segunda linha, respectivamente.

```
set eb [new Agent/Estacao_Base]
set v1 [new Agent/Veiculo]
$ns_ attach-agent $node_(0) $eb
$ns_ attach-agent $node_(1) $v1
$ns_ connect $eb $v1
```

Figura 33. Criação de agentes e ligação com os nós.

Na terceira linha, o ‘nó 0’ é ligado ao objeto ‘eb’ que é um agente ‘Estacao_Base’; na quarta linha, o ‘nó 1’ é ligado ao objeto ‘v1’ que é um agente ‘Veiculo’; e na última linha os dois objetos (‘eb’ e ‘v1’) são conectados.

Após a finalização desses passos os objetos podem se comunicar. Uma forma de realizar a troca de mensagens é através do escalonador de eventos, conforme mostra a Figura 34.

```
$ns_at 4.0 "$v1 sincronizar"
$ns_at 8.0 "$eb enviarMensagem"
```

Figura 34. Escalonador de eventos.

A primeira linha mostra o objeto ‘v1’ tentando sincronizar com a estação base no tempo 4.0 (relógio do simulador), enquanto a segunda linha mostra o objeto ‘eb’ enviando uma mensagem no tempo 8.0.

É importante mencionar que o simulador NS-2 oferece suporte para o padrão IEEE 802.11 – muito utilizado por pesquisadores de redes sem fio. Outra modelagem desse padrão foi desenvolvida por Chen et al. (2007) – descrita também em Chen et al. (2008) –, com a introdução de dois módulos novos: Mac802.11Ext e WirelessPhyExt.

Esses módulos têm o propósito de corrigir algumas deficiências da modelagem do padrão IEEE 802.11, como por exemplo, a mistura das funcionalidades das camadas física e lógica, a complexidade excessiva da camada lógica e a simplificação ou imprecisão na modelagem desse padrão (CHEN et al., 2008).

Além desses dois módulos, Chen et al. (2008) alertam que os valores observados na Figura 35 também devem ser adicionados ao *script* de simulação para manter uma maior precisão nos resultados das simulações em redes veiculares.

```
# 802.11p default parameters

Phy/WirelessPhyExt set CSThresh_                    3.9810717055349694e-13
Phy/WirelessPhyExt set Pt_                          0.1
Phy/WirelessPhyExt set freq_                        5.9e+9
Phy/WirelessPhyExt set noise_floor_                 1.26e-13
Phy/WirelessPhyExt set L_                           1.0
Phy/WirelessPhyExt set PowerMonitorThresh_          3.981071705534985e-18
Phy/WirelessPhyExt set HeaderDuration_              0.000040
Phy/WirelessPhyExt set BasicModulationScheme_       0
Phy/WirelessPhyExt set PreambleCaptureSwitch_       1
Phy/WirelessPhyExt set DataCaptureSwitch_           1
Phy/WirelessPhyExt set SINR_PreambleCapture_        3.1623
Phy/WirelessPhyExt set SINR_DataCapture_            10.0
Phy/WirelessPhyExt set trace_dist_                  1e6
Phy/WirelessPhyExt set PHY_DBG_                     0

Mac/802_11Ext set CWMin_                            15
Mac/802_11Ext set CWMax_                            1023
Mac/802_11Ext set SlotTime_                         0.000013
Mac/802_11Ext set SIFS_                             0.000032
Mac/802_11Ext set ShortRetryLimit_                  7
Mac/802_11Ext set LongRetryLimit_                   4
Mac/802_11Ext set HeaderDuration_                   0.000040
Mac/802_11Ext set SymbolDuration_                   0.000008
Mac/802_11Ext set BasicModulationScheme_            0
Mac/802_11Ext set use_802_11a_flag_                 true
Mac/802_11Ext set RTSThreshold_                     2346
Mac/802_11Ext set MAC_DBG                           0
```

Figura 35. Parâmetros de configuração do padrão 802.11p.

O código da Figura 35 (e uma documentação complementar) é disponibilizado pelo grupo de pesquisa desenvolvedor dos módulos Mac802.11Ext e WirelessPhyExt, e fornece os parâmetros necessários para a simulação do padrão IEEE 802.11p que garante acesso sem fio em ambientes veiculares (KARLSRUHE INSTITUTE OF TECHNOLOGY, [2014?]).

Esses valores regulam a capacidade dos veículos de enviar e receber os pacotes de dados, configurando, por exemplo, o poder de transmissão (Pt_), o número de tentativas de retransmissão (ShortRetryLimit_ and LongRetryLimit_), detecção de portadora (CSThresh_) e a ação de habilitar/desabilitar o mecanismo usado pra evitar colisões de pacotes (RTSThreshold_).

6.2 Viabilidade do Protocolo µTESLA

Este teste exigiu uma configuração específica para a criação do cenário, além das configurações mencionadas acima. Sendo assim, uma estação base foi colocada a 10 metros

de distância da estrada e o alcance de transmissão da antena da estação base foi limitado a 40 metros.

Considerando que um veículo está trafegando a uma velocidade média de 120 km/h, em linha reta, levará cerca de 2,32 segundos para que seja percorrida uma distância de aproximadamente 77,46 metros referente ao percurso entre o ponto A e o ponto B, conforme ilustrado na Figura 36.

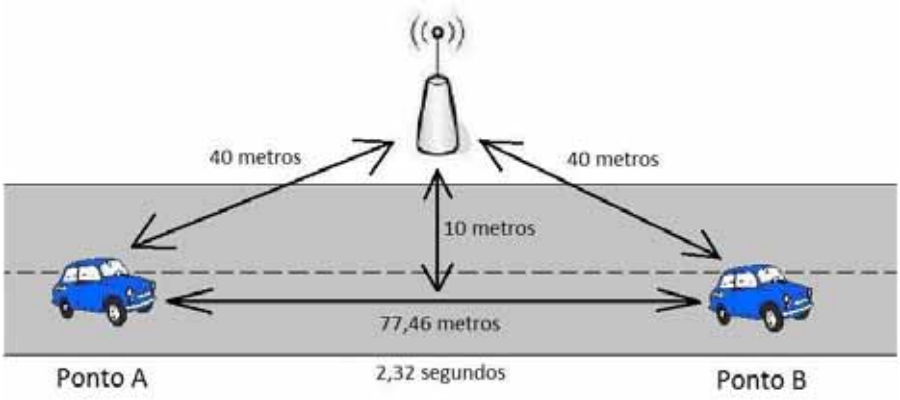


Figura 36. Intervalo de transmissão.

Esses dois pontos (A e B) correspondem ao limiar de transmissão da antena da estação base, isto é, no ponto A, o veículo começa a receber mensagens e após o ponto B, isso não é mais possível.

Essas configurações caracterizam uma situação limite, mostrando o correto funcionamento do μTESLA em um cenário de pouca distância e curto espaço de tempo, ressaltando a rapidez na comunicação.

Com relação ao protocolo, as configurações usadas foram de 0,05 segundo para cada intervalo de tempo e as chaves de autenticação foram reveladas a cada 2 intervalos de tempo. Ou seja, uma chave é enviada a cada 0,1 segundo, possibilitando a autenticação de diversas mensagens.

Nessas condições o veículo gastou 0,0022 segundo para sincronizar com a estação base. Após isso, conseguiu autenticar mensagens em um tempo de 0,075 segundo, em média.

Sendo assim, o veículo consegue autenticar cerca de 30 conjuntos de mensagens dentro de um intervalo de tempo de 2,32 segundos. É importante observar que com uma única chave o veículo autentica todas as mensagens enviadas dentro de um mesmo intervalo de tempo, isto é, cada chave autentica um conjunto de mensagens e não apenas uma.

6.3 Comparação entre os Protocolos µTESLA e CMAP

Uma maneira clara de se verificar a viabilidade de um protocolo qualquer é a comparação com outro protocolo de mesma categoria. Escolhemos, então, o protocolo CMAP (HAO et al., 2011) para confrontar seus resultados com os resultados obtidos com o µTESLA.

O CMAP é um protocolo que tem o mesmo objetivo do µTESLA, ou seja, a autenticação de mensagens enviadas por radiodifusão. Contudo, seu funcionamento é diferente, uma vez que se utiliza de assinatura digital e criptografia para autenticar as mensagens.

Trabalhando de forma cooperativa, o CMAP pretende diminuir o número de mensagens de segurança que os veículos precisam receber, compartilhando os resultados de verificação obtidos por eles e, por conseguinte, reduzindo a sobrecarga da rede.

Duas comparações foram realizadas entre esses protocolos, sendo que a primeira tem o objetivo de medir a eficiência e a segunda, a escalabilidade.

A) Eficiência

A eficiência de um protocolo diz respeito à rapidez, isto é, o quão rápido é o protocolo no momento de transmitir uma mensagem autenticada de um nó a outro.

De acordo com as simulações realizadas, um veículo precisou esperar cerca de 0,05 segundo, no melhor caso, e cerca de 0,1 segundo, no pior caso, para conseguir usar uma mensagem autenticada usando o protocolo µTESLA. Ou seja, um veículo levou, em média, 0,075 segundo para autenticar uma mensagem, com um desvio padrão de 0,035.

Nas mesmas condições, um veículo esperou cerca de 0,23 segundo, no melhor caso, e cerca de 0,33 segundo, no pior caso, usando o protocolo CMAP. Ou seja, um veículo precisou de 0,28 segundo, em média, para autenticar uma mensagem, com um desvio padrão de 0,07.

Na Figura 37 pode-se observar o tempo médio gasto para o recebimento de uma mensagem autenticada por cada protocolo.

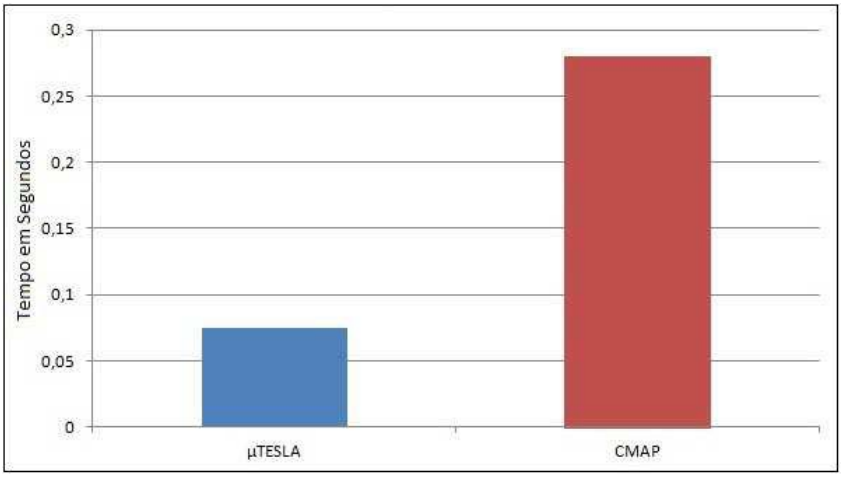


Figura 37. Tempo médio de recebimento de mensagens.

Esses resultados mostram que o protocolo µTESLA, apesar de seu atraso por conta da divulgação tardia das chaves de autenticação, é bastante eficiente se comparado ao CMAP.

A vantagem do µTESLA se explica devido ao uso de assinatura digital (ECDSA) e criptografia assimétrica (ECIES) pelo CMAP. A criptografia assimétrica é mais lenta que a criptografia simétrica, pois se utiliza de algoritmos mais complexos, exigindo muito mais poder de processamento.

A despeito disso, o protocolo CMAP utiliza-se de muitas operações de criptografia (cifragem e decifração) e assinatura (assinatura e verificação) em todo o seu processo de comunicação e autenticação de mensagens, causando um atraso maior na rede.

B) Escalabilidade

A escalabilidade é outra característica importante nas redes veiculares, pois a quantidade de veículos em uma determinada rede pode ser bastante grande, como, por exemplo, em um congestionamento ou em um horário de pico.

Em um cenário com uma maior densidade de veículos, a tendência é diminuir o desempenho da rede, pois, a quantidade de veículos causa um impacto sobre a conectividade da rede e sobre a probabilidade de congestionamento no canal sem fio.

Na Figura 38 é mostrado o desempenho dos protocolos µTESLA e CMAP com relação à perda de pacotes à medida que aumenta o número de veículos na rede. As tabelas com os dados completos das simulações são apresentadas no apêndice A.

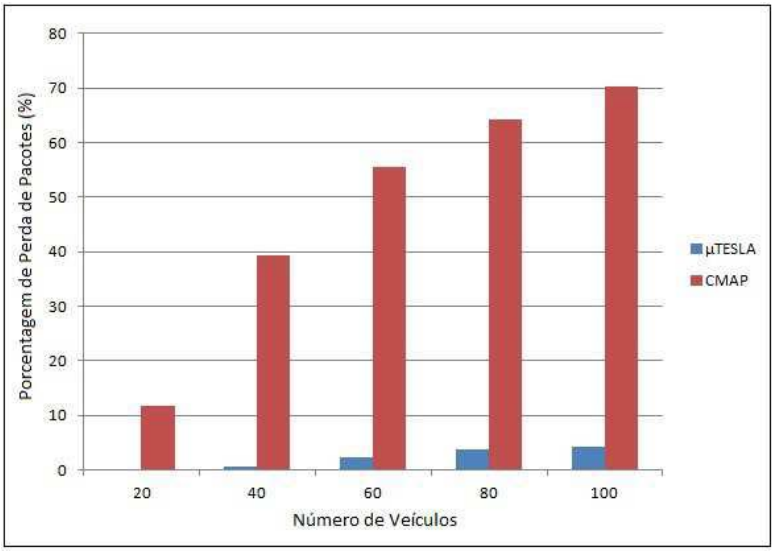


Figura 38. Perda de pacotes.

A quantidade de pacotes perdidos pelo protocolo µTESLA foi muito inferior ao observado com CMAP. Pode-se observar, inclusive, que a perda de pacotes cresce pouco conforme aumenta a quantidade de veículos na rede, sendo que essa perda é praticamente inexistente (menos de 0,1%) no caso de haver até 20 veículos.

O mesmo não acontece com o protocolo CMAP, isto é, pode-se observar um crescimento muito significativo de perda de pacotes conforme aumenta a quantidade de veículos na rede. Uma explicação para essa diferença é a maior sobrecarga na rede causada pelo protocolo CMAP, apesar de sua proposta de diminuir tal sobrecarga.

O CMAP exige muitas trocas de mensagens: primeiramente, entre a estação base e a autoridade certificadora, para recebimento de chaves de segurança e certificado; depois, entre o veículo e a estação base, para efetivar o processo de registro; por fim, entre os veículos emissor e receptor, para a troca de informações. Essas trocas de mensagens – somando-se com outras de roteamento – geram um tráfego de rede muito intenso causando uma grande perda de pacotes.

6.4 Considerações Finais

Neste capítulo foram apresentados os resultados alcançados neste trabalho. Para que as simulações dos protocolos gerassem resultados de forma mais precisa possível, foi simulado um modelo de mobilidade no VanetMobiSim.

Esse modelo permitiu a criação de cenários com várias faixas, múltiplos fluxos convergentes e cruzamentos regulados por semáforos e por sinais de parada. Os veículos ainda conseguiam mudar de faixa e fazer ultrapassagens, sempre de acordo com as normas de trânsito.

O arquivo de saída com o padrão de mobilidade para os veículos do VanetMobisim foram usados no NS-2. Para o uso deste segundo simulador, foi necessária a criação de um *script* com a configuração e a definição dos valores das variáveis de uma rede veicular.

Para testar a viabilidade do protocolo μTESLA foi criado um cenário em que um veículo passa por uma estação base transmitindo mensagens com um pequeno poder de transmissão. Do momento em que o veículo começa a receber as mensagens até o momento em que deixa de recebê-las passam-se 2,32 segundos. Os resultados das simulações mostraram que o veículo conseguiu autenticar cerca de 30 conjuntos de mensagens nesse intervalo de tempo.

Nas comparações realizadas entre os protocolos μTESLA e CMAP, foi mostrado que μTESLA foi quase 4 vezes mais rápido do que CMAP no quesito eficiência, devido a este último utilizar-se de criptografia assimétrica e assinatura digital para manter a comunicação entre os veículos e autenticar suas mensagens.

Com relação à escalabilidade, o desempenho do μTESLA também foi superior ao CMAP. Em uma rede com até 20 veículos, houve uma perda de pacote inferior a 0,1% pelo protocolo μTESLA, sendo que a porcentagem de pacotes perdidos, à medida que aumentou o número de veículos na rede, também foi pequena se comparada às perdas pelo protocolo CMAP.

7 CONCLUSÃO

Em 7.1 é feita uma análise dos resultados alcançados. Em 7.2 são apresentadas as dificuldades encontradas ao longo do trabalho. Por fim, em 7.3 são apresentadas as sugestões para trabalhos futuros.

7.1 Análise dos Resultados

Realizar qualquer tipo de teste em uma rede veicular é uma tarefa complexa e dispendiosa em um ambiente real. Como alternativa, os simuladores são ferramentas bastante utilizadas pelos pesquisadores e desenvolvedores para testar as novas tecnologias.

Diante disso, os protocolos μTESLA e CMAP foram implementados e comparados no NS-2 para que a proposta desse trabalho fosse devidamente validada.

Os resultados gerados por meio das simulações mostraram que o protocolo μTESLA superou CMAP em todos os testes.

O protocolo μTESLA mostrou-se mais eficiente do que CMAP. Essa vantagem se explica devido ao CMAP gastar tempo realizando operações de criptografia assimétrica e assinatura digital. De maneira mais simples, μTESLA autentica suas mensagens usando MACs e divulgando suas chaves de autenticação tardiamente.

μTESLA também se mostrou mais escalável em comparação ao CMAP. O sistema utilizado pelo CMAP exige uma grande quantidade de transmissão de mensagens, devido aos seus processos de registro de veículos, de acusação e de envios de certificado, de chaves, de mensagens regulares (RBM – *Regular Broadcast Message*) e de mensagens de aviso (CAM – *Cooperative Authentication Message*). Com isso, o tráfego na rede aumenta significativamente, acarretando mais colisões de pacotes e, consequentemente, a perda desses pacotes é maior.

Por sua vez, a porcentagem de perda de pacotes do μTESLA é pequena e varia pouco à medida que aumenta o número de veículos na rede. Isso ocorre porque cada veículo que entra na rede exige apenas duas mensagens adicionais, transmitidas no momento da sincronização. Depois disso, nenhuma mensagem adicional é necessária, pois cada veículo gerencia seu processo de autenticação das mensagens recebidas.

A partir desses resultados, consideramos que μTESLA é um protocolo seguro, eficiente e escalável, demonstrando alta viabilidade para ser usado nas redes veiculares.

7.2 Dificuldades Encontradas

A grande dificuldade encontrada em todo o projeto foi a compreensão e a utilização do NS-2. Apesar de existir uma grande documentação, entre *sites* e manuais, acredito que ainda falta muita coisa para ser explicada sobre esse simulador devido à sua enorme complexidade.

As dificuldades se deram desde a instalação do simulador, passando pelas simulações de teste e chegando até a programação, pois toda a implementação dos protocolos foi feita dentro do NS-2, uma vez que todos os módulos (os já existentes e os novos) precisam funcionar em conjunto.

7.3 Sugestões para Trabalhos Futuros

Como trabalho futuro, pretendemos criar novos cenários para medir o desempenho do protocolo μTESLA, utilizando, nas simulações, topologias que poderão ser criadas usando os padrões de mobilidade gerados a partir de mapas de cidades da região.

Pretendemos, também, fazer uma caracterização das diferentes classes de aplicações: segurança, informação e entretenimento. Diferentes cenários podem ser pensados para cada tipo de classe, como por exemplo:

a) Segurança no trânsito: um motorista na estrada recebe a informação para desviar seu caminho, pois tem um grande engarrafamento logo a sua frente;

b) Assistência ao motorista: um motorista com um passageiro passando mal recebe a informação da localização do hospital mais próximo;

c) Entretenimento: um passageiro (que pode ser uma criança) está compartilhando um vídeo e, de repente, começa a receber filmes com conteúdos inadequados.

Nas duas primeiras situações, o motorista pode estar sendo direcionado para algum beco sem saída ou qualquer outro local perigoso, onde pode ser alvo de pessoas mal intencionadas. Na terceira situação os danos são de difícil dimensionamento.

Contudo, nas três situações descritas acima é de suma importância conhecer a origem dos dados. Pois somente dessa forma, os possíveis danos podem ser evitados ou o emissor dos dados pode vir a ser responsabilizado por suas ações.

Pretendemos, ainda, simular possíveis ataques ao protocolo μTESLA, como transmissão de mensagens falsas, alteração das mensagens transmitidas, negação de serviço, replicação de chaves, entre outros. Com os resultados obtidos, é possível analisar o comportamento da rede e fazer um estudo dos possíveis impactos gerados por cada tipo de ataque.

REFERÊNCIAS

ALTMAN, E.; JIMÉNEZ, T. **NS simulator for beginners**. Mérida: Universidade de Los Andes; Sophia-Antipolis: ESSI, 2003.

CANU MOBILITY SIMULATION ENVIRONMENT (CanuMobiSim), [2014?]. Disponível em: <http://canu.informatik.uni-stuttgart.de/mobisim/>. Acesso em: 7 jun. 2014.

CAR 2 CAR COMMUNICATION CONSORTIUM (C2C-CC). **Mission and objectives**. Braunschweig, [2014?]. Disponível em: <http://www.car-2-car.org/>. Acesso em: 7 jun. 2014.

CHEN, Q. et al. Overhaul of IEEE 802.11 modeling and simulation in ns-2. In: **Proceedings of the 10th ACM Symposium on Modeling, Analysis, and Simulation of Wireless and Móbile Systems**, New York, 2007. p. 159-168.

CHEN, Q. et al. **Overhaul of IEEE 802.11 modeling and simulation in ns-2 (802.11Ext)**. Karlsrube: University of Karlsrube; Sunnyvale: Mercedes-Benz Research & Development North America, 2008.

CHUNG, J; CLAYPOOL, M. **NS by example**. Worcester: Worcester Polytechnic Institute. Computer Science Department, [2014?]. Disponível em: <http://nile.wpi.edu/NS/>. Acesso em: 7 jun. 2014.

ERTICO ITS EUROPE. **Intelligent transport systems and services for Europe** – bringing intelligence into mobility for people and goods across Europe. [2014?]. Disponível em: <http://www.ertico.com/ertico-its-europe/>. Acesso em: 7 jun. 2014.

EVITA. **Objectives**, [2008?]. Disponível em: <http://www.evita-project.org/>. Acesso em: 7 jun. 2014.

FIORE, M. et al. Vehicular mobility simulation for VANETs. In**: 40th Annual Simulation Symposium,** Norfolk: IEEE, 2007**.** p. 301-309.

GROOVENET. **A vehicular network virtualization platform**, [2014?]. Disponível em: <http://mlab.seas.upenn.edu/groovenet/>. Acesso em: 7 jun. 2014.

HAO, Y et al. A distributed key management framework with cooperative message authentication in VANETs. In: **IEEE Journal on selected areas in communications**, v. 29, n. 3, Mar. 2011. p. 616-629.

HAO, Y.; HAN, T.; CHENG, Y. A cooperative message authentication protocol in VANETs. In: **Global Communications Conference (GLOBECOM) 2012**, Anaheim: IEEE, 2012. p. 5562-5566.

HÄRRI, J; FIORE, M. **VanetMobiSim – vehicular ad hoc network mobility extension to the CanuMobiSim framework**. Torino: Institut Eurécom, Politecnico di Torino, 2005/06. Manual.

HASSAN, A. **VANET simulation**. 2009. 43 f. Dissertação (Mestrado em Engenharia Elétrica), School of Information Science, Computer and Electrical Engineering. Halmstad University, 2009.

HOGIE, L.; BOUVRY, P.; GUINAND, F. An overview of MANETs simulation. **Journal Electronic Notes in Theoretical Computer Science (ENTCS)**, v. 150, issue 1, Mar. 2006. p. 81-101.

INSTITUTO NACIONAL DE CIÊNCIA E TECNOLOGIA EM SISTEMAS EMBARCADOS CRÍTICOS (INCT-SEC). **CARINA**, [2014?]. Disponível em: <http://www.inct-sec.org/br/aplicacoes/carina>. Acesso em: 12 ago. 2014.

ITS AMERICA. **Technology transforming transportation**, [2014?]. Disponível em: <http://www.itsa.org/>. Acesso em: 7 jun. 2014.

JABALLAH, W. B. et al. Modeling source authentication protocols in wireless sensor networks using HLPSL. In: **Conference on Network and Information Systems Security (SAR-SSI)**, La Rochelle: IEEE, May. 2011. p. 1-9.

JAPAN AUTOMOBILE RESEARCH INSTITUTE (JARI). **About JARI**, [2014?]. Disponível em: <http://www.jari.or.jp/tabid/200/Default.aspx?language=en-US>. Acesso em: 7 jun. 2014.

KARLSRUHE INSTITUTE OF TECHNOLOGY. **Decentralized Systems and Network Services Research Group – TM & SCC**, [2014?]. Disponível em: <http://dsn.tm.kit.edu/english/Overhaul_NS-2.php>. Acesso em: 7 jun. 2014.

KESHAV, S. **REAL 5.0 overview**, Cornell University, Aug. 1997. Disponível em: <http://www.cs.cornell.edu/skeshav/real/overview.html>. Acesso em: 7 jun. 2014.

LIN, X.; LI, X. Achieving efficient cooperative message authentication in vehicular ad hoc networks. **IEEE Transactions on Vehicular Technology**, v. 62, issue 7, p. 3339-3348, Sept. 2013.

LU, Y. et al. Group-based secure source authentication protocol for VANETs. In: **Globecom 2010 Workshop on Heterogeneous, Multi-hop Wireless and Mobile Networks**, Miami: IEEE, Dec. 2010. p. 202-206.

LYU, C. et al. Efficient, fast and scalable authentication for VANETs. In: **Wireless Communications and Networking Conference (WCNC)**, Shanghai: IEEE, Apr. 2013. p. 1768-1773.

MARTINEZ, F. J. et al. A survey and comparative study of simulators for vehicular ad hoc networks (VANETs). In: **Wireless Communications and Mobile Computing**, v. 11, issue 7, July 2009. p. 813-828.

MOBIREAL. **A realistic network simulator**, [2014?]. Disponível em: <http://www.mobireal.net/>. Acesso em: 7 jun. 2014.

MONARCH PROJECT. **Wireless and mobility extensions to NS-2**, [2014?]. Disponível em: <http://www.monarch.cs.rice.edu/cmu-ns.html>. Acesso em: 7 jun. 2014.

NCTUNS 6.0. **Network simulator and emulator**, [2014?]. Disponível em: <http://nsl.csie.nctu.edu.tw/nctuns.html>. Acesso em: 7 jun. 2014.

NETWORK TIME PROTOCOL. **O NTP**, [2014?]. Disponível em: <http://www.ntp.br/NTP/MenuNTPNtp>. Acesso em: 12 dez. 2014.

OLIVEIRA, R. Introdução ao simulador de redes "The Network Simulator NS-2". **Mobilidade em Sistemas de Telecomunicações**. Faculdade de Ciências e Tecnologia. Universidade Nova de Lisboa, 2011.

PERRIG, A. et al. Efficient authentication and signing of multicast streams over lossy channels. In: **IEEE Symposium on Security and Privacy**, Berkeley. IEEE, May. 2000. p. 56-73.

PERRIG, A. et al. SPINS: Security protocols for sensor networks. Berkeley: University of California, Department of Electrical Engineering and Computer Sciences. In: **Mobile Computing and Networking**, 2001.

PIÓRKOWSKI, M. et al. TraNS: Realistic Joint Traffic and Network Simulator for VANETs. School of Computer and Communication Sciences EPFL, Switzerland. **ACM SIGMOBILE Mobile Computing and Communications Review**, v. 12, 1, p. 31-33, Jan. 2008.

RAFIQ, G. et al. What's new in intelligent transportation systems?: an overview of european projects and initiatives. **IEEE Vehicular Technology Magazine**, v. 8, p. 45-69, Dec. 2013.

RAYA, M.; HUBAUX, J.-P. Securing vehicular ad hoc networks. **Journal of Computer Security**, v. 15, p. 39-68, Jan. 2007.

RAYA, M.; PAPADIMITRATOS, P.; HUBAUX, J.-P. Securing vehicular communications. **IEEE Wireless Communications**, v. 13, p. 8-15, Oct. 2006.

RESEARCH AND INNOVATIVE TECHNOLOGY ADMINISTRATION (RITA). **ITS Joint Program Office – mission**, 2013. Disponível em: <http://www.standards.its.dot.gov/Factsheets/Factsheet/80>. Acesso em: 7 jun. 2014.

STALLINGS, W. Autenticação de mensagem e funções de hash. In:_____. **Criptografia e segurança de redes:** princípios e práticas. Tradução Daniel Vieira; Revisão técnica Ákio Barbosa e Marcelo Succi. 4. ed. São Paulo: Pearson Prentice Hall, 2008. p. 226-251.

STUDER, A. et al. Flexible, extensible, and efficient VANET authentication. **Journal of Communications and Networks,** v. 11, 6, Dec. 2009. p. 574-588.

THE NETWORK SIMULATOR (NS-2), [2014?]. Disponível em: <http://www.isi.edu/nsnam/ns/>. Acesso em: 7 jun. 2014.

WANG, M. et al. Modeling and analyzing the μTESLA protocol using CSP. In: **5th International Symposium on Theoretical Aspects of Software Engineering (TASE)**, Xi'an: IEEE. Aug. 2011. p. 247-250.

YEO, D.; YOUM, H. An μTESLA protocols with multi-senders based on a 2-level XOR chain with data-loss tolerance. In: **10th Annual IEEE/IPSJ International Symposium on Applications and the Internet (SAINT)**, Seoul: IEEE. July 2010. p. 269-272.

ZEADALLY, S. et al. Vehicular ad hoc networks (VANETS): status, results, and challenges. **Telecommunication Systems,** v. 50, 4, Aug. 2010. p. 217-241.

APÊNDICE A – Tabelas com os dados das simulações

A.1 Dados do protocolo μTESLA

Quantidade de Veículos	Pacotes Enviados	Pacotes Recebidos	Pacotes Perdidos	Porcentagem de Perda (%)
10	25.420	25.417	3	0,01
20	53.220	53.189	31	0,06
30	84.213	83.881	332	0,39
40	117.639	116.871	768	0,65
50	153.202	150.051	3.151	2,06
60	192.473	188.109	4.364	2,27
70	232.446	225.076	7.370	3,17
80	275.134	264.968	10.166	3,69
90	323.314	310.682	12.632	3,91
100	371.352	355.525	15.827	4,26

A.2 Dados do protocolo CMAP

Quantidade de Veículos	Pacotes Enviados	Pacotes Recebidos	Pacotes Perdidos	Porcentagem de Perda (%)
10	45.988	45.270	718	1,56
20	174.361	153.976	20.385	11,69
30	385.903	281.046	104.857	27,17
40	680.519	413.114	267.405	39,29
50	1.057.519	540.797	516.722	48,86
60	1.516.835	673.754	843.081	55,58
70	2.060.800	825.099	1.235.701	59,96
80	2.686.417	957.851	1.728.566	64,34
90	3.395.214	1.099.641	2.295.573	67,61
100	4.183.689	1.242.604	2.941.085	70,3

Autorizo a reprodução xerográfica para fins de pesquisa.

São José do Rio Preto, 29 / 12 / 2014



Assinatura