



**UNIVERSIDADE FEDERAL DE SERGIPE**  
**PRÓ-REITORIA DE PÓS-GRADUAÇÃO E PESQUISA**  
**NÚCLEO DE PÓS-GRADUAÇÃO EM CIÊNCIA DA**  
**COMPUTAÇÃO**

**GEORGE LEITE JUNIOR**

**I9Vanets: um modelo de arquitetura de  
software para rede veicular em nuvem**

**São Cristóvão**

**2017**

**GEORGE LEITE JUNIOR**

**I9Vanets: um modelo de arquitetura de software para  
rede veicular em nuvem**

Versão original

Dissertação apresentada à Pró-Reitoria de Pós-Graduação e Pesquisa da Universidade Federal de Sergipe para obtenção do título de Mestre em Ciência da Computação pelo Programa de Pós-graduação em Ciência da Computação.

Área de concentração: Redes de Computadores e Sistemas Distribuídos

Orientador: Prof.<sup>a</sup> Dr.<sup>a</sup> Douglas D. J. de Macedo

Coorientador: Prof. Dr. Rogério Patrício Chagas do Nascimento

São Cristóvão

2017

Dissertação de autoria de George Leite Junior, sob o título **“I9Vanets: um modelo de arquitetura de software para rede veicular em nuvem”**, apresentada à Pró-Reitoria de Pós-Graduação e Pesquisa da Universidade Federal de Sergipe, para obtenção do título de Mestre em Ciência da Computação pelo Programa de Pós-graduação em Ciência da Computação, na área de concentração Redes de Computadores e Sistemas Distribuídos, aprovada em \_\_\_\_ de \_\_\_\_\_ de \_\_\_\_\_ pela comissão julgadora constituída pelos doutores:

**Prof.<sup>a</sup> Dr.<sup>a</sup> Douglas D. J. de Macedo**

Presidente

Instituição: Universidade Federal de Santa Catarina

**Prof. Dr. Rogério Patrício Chagas do Nascimento**

Instituição: Universidade Federal de Sergipe

**Prof. Dr.** \_\_\_\_\_

**Instituição:** \_\_\_\_\_

## Resumo



LEITE, George Junior. **i9Vanets: um modelo de arquitetura de software para rede veicular em nuvem**. 2017. 90 f. Dissertação (Mestrado em Ciência da Computação) – Pró-Reitoria de Pós-Graduação e Pesquisa, Universidade Federal de Sergipe, Sergipe, 2016.


Em consequência do crescimento populacional, as grandes cidades enfrentam problemas cotidianos relacionados à mobilidade urbana tais como: congestionamentos, baixa qualidade das rodovias, ineficiência de transportes públicos, entre outros. Iniciativas de sistemas de transportes inteligentes (STI) agem como uma solução eficiente para melhorar o funcionamento e desempenho dos sistemas de tráfego, reduzindo congestionamentos e aumentando a segurança para os cidadãos. Atualmente, pesquisadores vem buscando nas redes veiculares ad-hoc (VANET) uma possível solução para os problemas referentes à mobilidade urbana. Contudo, VANETs ainda apresenta uma série de desafios que devem ser resolvidos para que seu uso seja consolidado. Deste modo, o presente trabalho **traz** uma arquitetura denominada i9Vanets, cujo intuito é o gerenciamento virtualizado por meio de uma rede veicular para auxiliar nas soluções dos **principais desafios relacionados à VANETs**. Também é apresentado uma análise, dos testes realizados em laboratório, com objetivo de avaliar seu desempenho e capacidade operacional. **Sendo assim, a conclusão deste trabalho, foi apresentar a viabilidade técnica da arquitetura i9Vanets bem como suas possíveis aplicações.**



Palavras-chaves: Rede Veicular. Computação em Nuvem e Arquitetura Distribuída.

## Abstract

LEITE, George Junior. **I9Vanets: a software architecture model for a vehicular network**. 2017. 90 p. Dissertation (Master of Science of Computer) – Dean of Graduate Studies and Research, Federal University of Sergipe, Sergipe, 2016.



As a result of population growth, large cities face daily problems related to urban mobility such as congestion, poor quality of roads, inefficiency of public transportation, among others. Intelligent Transport Systems (ITS) initiatives act as an efficient solution to improve the operation and performance of traffic systems, reducing traffic jam and increasing safety for citizens. Currently, researchers have been looking at ad-hoc vehicular networks (VANET) as a possible solution to problems related to urban mobility. However, VANETs still presents a number of challenges that must be addressed in order for its use to be consolidated. Thus, the present work brings an architecture called i9Vanets, whose purpose is the virtualized management through a vehicular network to assist in the solutions of the main challenges related to VANETs. We also present an analysis of the tests carried out in the laboratory, in order to evaluate their performance and operational capacity. Therefore, the conclusion of this work was to present the technical feasibility of the i9Vanets architecture as well as its possible applications.

Keywords: Vehicle Network. Cloud Computing and Distributed Architecture .

## Lista de figuras

Figura 1 – Imagem do trânsito (DANTAS, 2011) . . . . .	14
Figura 2 – Modelo do VNC (LEE et al., 2014) . . . . .	18
Figura 3 – Arquitetura de referência para aplicações ITS baseadas em WSN. (LO-SILLA et al., 2011) . . . . .	23
Figura 4 – Geração de componentes de cliente e servidor, a partir da interface para Web Services, CORBA e Java-RMI (GRAY, 2004) . . . . .	33
Figura 5 – Etapas de uma conexão <i>WebSocket</i> (THEMUDO, 2014) . . . . .	34
Figura 6 – HTTP, HTTPS, WS e WSS (THEMUDO, 2014) . . . . .	35
Figura 7 – Arquiteturas de redes veiculares(LUÍS, 2009) . . . . .	37
Figura 8 – Ataque DDOS (WANGHAM et al., 2014). . . . .	39
Figura 9 – Ataque Buraco Negro (WANGHAM et al., 2014). . . . .	40
Figura 10 – Ataque social (WANGHAM et al., 2014). . . . .	41
Figura 11 – Ataque demodificação de mensagem (WANGHAM et al., 2014). . . . .	42
Figura 12 – Exemplo de aplicação VANET (QIAN; LU; MOAYERI, 2008). . . . .	50
Figura 13 – Módulos definidos para a arquitetura I9Vanet. . . . .	51
Figura 14 – Modelo de comunicação V2A, I2A, AI2AI e AV2AI. . . . .	54
Figura 15 – Exemplo de domínios. . . . .	59
Figura 16 – Organização dos servidores na nuvem. . . . .	61
Figura 17 – Modelo de processo de negócio da arquitetura I9Vanet com criptografia. . . . .	63
Figura 18 – Modelo de processo de negócio da arquitetura I9Vanet sem criptografia. . . . .	64
Figura 19 – Coniguração do ambiente utilizado para a realização dos experimentos. Fonte: Criada pelo autor. . . . .	68
Figura 20 – Tela de monitoramento do Sistema TaxiFast. . . . .	68
Figura 21 – Análise de Histograma de Frequência do Teste com 50 veículos com comunicação aberta. . . . .	74
Figura 22 – Análise de Histograma de Frequência do Teste com 100 veículos com comunicação aberta. . . . .	75
Figura 23 – Análise de Histograma de Frequência do Teste com 200 veículos com comunicação aberta. . . . .	76
Figura 24 – Análise de Histograma de Frequência do Teste com 400 veículos com comunicação aberta. . . . .	76

Figura 25 – Gráfico comparativo dos tempos mínimos entres os cenários. . . . .	77
Figura 26 – Gráfico comparativo dos tempos médios entres os cenários. . . . .	77
Figura 27 – Gráfico comparativo dos tempos máximos entres os cenários. . . . .	77
Figura 28 – Comparação dos gráficos de histograma para os RTTs com 800 veículos.	78
Figura 29 – Comparação dos gráficos de histograma para os RTTs com 1600 veículos.	78
Figura 30 – Gráfico comparativo dos tempos mínimos entres os cenários 800 e 1600.	79
Figura 31 – Comparativo da capacidade de cada <i>link</i> pela quantidade de veículos. .	80
Figura 32 – Nível de processamento dos servidores para o teste com 800 veículos com conteúdo criptografado. . . . .	80
Figura 33 – Nível de processamento dos servidores para o teste com 1600 veículos com conteúdo criptografado. . . . .	81
Figura 34 – Percentual de perda para cada experimento realizado.. . . .	81

## Lista de algoritmos



## Lista de tabelas

Tabela 1 – Análise de custos de solicitação de uma única requisição utilizando várias tecnologias(GRAY, 2004) . . . . .	33
Tabela 2 – Protocolos de roteamento aplicados a redes veiculares (LUÍS, 2009) . .	49
Tabela 3 – Quadro de análise sobre os tipos de ataques e a arquitetura I9Vanets. .	58
Tabela 4 – Colunas das tabelas <i>Server</i> . . . . .	60
Tabela 5 – Colunas da tabela <i>Device</i> . . . . .	60
Tabela 6 – Características das aplicações veiculares (PAPADIMITRATOS et al., 2008). . . . .	66
Tabela 7 – Linhas do arquivo de movimentação. . . . .	69
Tabela 8 – Configuração dos computadores utilizados no experimento. . . . .	71
Tabela 9 – Configuração dos computadores utilizados no experimento em cada cenário. . . . .	72
Tabela 10 – Resultado do teste de análise de distribuição normal dos dados para cada quantidade de veículos e velocidades de acesso. . . . .	73
Tabela 11 – Resultado do teste de análise dos coeficientes de variação com mensagens abertas. . . . .	73
Tabela 12 – Resultado do teste de análise dos coeficientes de variação com mensagens criptografadas. . . . .	74
Tabela 13 – Velocidade de cada link por veículo definidos de acordo com Li et al. (2009). . . . .	79

## Lista de abreviaturas e siglas

ITS	Intelligent Transport System
VANET	Vehicle Network Ad-hoc
VCC	Vehicle Cloud Computing
VNC	Vehicular Cloud Networking
V2V	Veículo para Veículo
V2I	Veículo para Infra-estrutura
AV2AV	Agente do Veículo para outro Agente do Veículo
AV2AI	Agente do Veículo para Agente da Infra-estrutura
AI2AI	Agente da Infra-estrutura para outro Agente da Infra-estrutura
AVC	Nuvens Veiculares Autônomas
SDN	Software-Defined Networking

## Sumário

1	Introdução . . . . .	13
1.1	Justificativa, Problemática e Hipótese . . . . .	14
1.1.1	Justificativa . . . . .	14
1.1.2	Problemática . . . . .	18
1.1.3	Hipótese . . . . .	18
1.2	Objetivos da Dissertação . . . . .	18
1.2.1	Objetivos Específicos . . . . .	19
1.2.2	Metodologia . . . . .	19
1.2.3	Contribuições . . . . .	20
1.3	Organização do Trabalho . . . . .	20
2	Rede de Sensores sem Fio . . . . .	22
2.1	Definição . . . . .	22
2.2	Arquitetura de Rede e Topologia . . . . .	22
2.2.1	Subsistema de Sensores . . . . .	23
2.2.2	Subsistema de Distribuição . . . . .	24
2.2.3	Subsistema de Tomada de Decisão . . . . .	25
2.2.4	Subsistema de Execução . . . . .	26
2.3	Considerações Finais do Capítulo . . . . .	27
3	Sistemas Distribuídos . . . . .	28
3.1	Definição . . . . .	28
3.2	Características e Desafios . . . . .	28
3.2.1	Heterogeneidade . . . . .	29
3.2.2	Sistemas Abertos . . . . .	29
3.2.3	Segurança . . . . .	29
3.2.4	Escalabilidade . . . . .	29
3.2.5	Tolerância a Falhas . . . . .	30
3.2.6	Transparência . . . . .	30
3.3	Comunicação em Sistemas Distribuídos . . . . .	31
3.3.1	Comunicação Cliente-Servidor . . . . .	31

3.3.2	Comunicação em Grupo . . . . .	32
3.3.3	Protocolos de Comunicação . . . . .	32
3.3.3.1	WebSocket . . . . .	34
3.4	Considerações Finais do Capítulo . . . . .	35
4	Redes Veiculares . . . . .	36
4.1	Definição . . . . .	36
4.2	Características e Desafios . . . . .	37
4.3	Segurança . . . . .	38
4.3.1	Ataques contra a Disponibilidade . . . . .	38
4.3.2	Ataques contra a Autenticidade e a Identificação . . . . .	40
4.3.3	Ataques contra a Integridade e Confiança dos Dados . . . . .	40
4.3.4	Ataques contra a Confidencialidade . . . . .	42
4.3.5	Outros Ataques . . . . .	42
4.4	Algoritmos de Roteamento . . . . .	43
4.4.1	Protocolos Ad-hoc . . . . .	44
4.4.2	Protocolos Baseados em Localização . . . . .	45
4.4.3	Protocolos Baseado em <i>Clusters</i> . . . . .	45
4.4.4	Protocolos por Broadcast . . . . .	46
4.4.5	Protocolos <i>Geocast</i> . . . . .	47
4.4.6	Comparação dos Protocolos de Roteamentos . . . . .	48
4.5	Aplicações . . . . .	49
4.6	Considerações Finais do Capítulo . . . . .	50
5	Arquitetura I9Vanet . . . . .	51
5.1	Visão Geral . . . . .	51
5.2	Módulos da Arquitetura I9Vanets . . . . .	51
5.2.1	Módulo de Comunicação . . . . .	53
5.2.1.1	Comunicação Infra-Cloud . . . . .	53
5.2.1.2	Comunicação Veículo-Cloud . . . . .	54
5.2.2	Módulo de Segurança . . . . .	55
5.2.3	Módulo de Gerenciamento dos Servidores . . . . .	58
5.2.4	Módulo de Roteamento . . . . .	62

5.2.5	Módulo de Aplicações . . . . .	62
5.3	Processo de Negócio . . . . .	63
5.4	Considerações Finais do Capítulo . . . . .	64
6	Avaliação da Arquitetura . . . . .	65
6.0.1	Definição . . . . .	65
6.0.2	Planejamento . . . . .	65
6.1	Cenário Proposto . . . . .	67
6.2	Experimentos . . . . .	71
6.3	Resultados . . . . .	72
6.3.1	Taxa de Transferência . . . . .	79
6.3.2	Processamento . . . . .	79
6.3.3	Perda de Pacotes . . . . .	80
7	Conclusão e Trabalhos Futuros . . . . .	82
7.1	Conclusões deste Trabalho . . . . .	82
7.2	Trabalhos Futuros . . . . .	83
	Referências . . . . .	84

# 1 Introdução

A conectividade inerente às cidades inteligentes abre uma fronteira bastante promissora no que se refere ao controle de acesso às informações (LI, 2010) (ZHU; LIU, 2015) e arquiteturas distribuídas para sistemas inteligentes de transporte (ICE et al., 2001).

Um sistema inteligente de transporte ou *Intelligent Transportation System* (ITS) representa “a aplicação de sensores avançados, computadores, dispositivos eletrônicos e tecnologias de comunicação e gerenciamento estratégico integrado visando melhorar a segurança e a eficiência do sistema de gerenciamento de tráfego” (NASIM; KASSLER, 2012).

Os Sistemas Inteligentes de Transporte são atualmente as mais importantes aplicações de uma rede veicular, fornecendo serviços de segurança rodoviária (XU et al., 2003) ou informações relativas às situações de tráfego. Dito por Sumra, Hasbullah e Manan (2011) e Luís (2009), o principal objetivo de uma VANET (*Vehicular ad-hoc Network*) é prover segurança aos passageiros nas estradas.

De acordo com Ball e Dulay (2010) e Losilla et al. (2011), a segurança no trânsito é a motivação principal para as pesquisas no âmbito das redes veiculares. Da mesma maneira, conforme a Organização Mundial da Saúde (OMS), acidentes acometidos por veículos são responsáveis por mais de um milhão de mortes e 50 milhões de feridos anualmente em todo o mundo (PEDEN et al., 2004). Nos Estados Unidos e no Brasil, acidentes relacionados a veículos são a terceira maior causa de mortalidade evitável e incapacitação profissional precoce (SYSTEMATICS; MEYER, 2011) (DIB et al., 2007).

Ainda afirmado por Losilla et al. (2011), os congestionamentos também são uma grande preocupação como mostra a figura 1. Só nos EUA, o congestionamento representa 115 bilhões de dólares em custos de combustível (TTI, 2014), com números semelhantes em outros países desenvolvidos. As baixas de tráfego no mundo ascendem a 1,17 milhões por ano. Neste contexto, os Sistemas de Transporte Inteligentes visam melhorar a eficiência e a segurança dos transportes através do uso avançado de processamento de informação, comunicação, controle, bem como, o uso de novas tecnologias.

Segundo o IPEA (Instituto de Pesquisa Econômica Aplicada do Brasil), a mobilidade urbana, nos dias atuais, é um dos principais problemas dos grandes centros. Os reflexos sobre o transporte urbano são evidentes, caracterizados principalmente pelo aumento do tráfego nas vias e consequentemente dos congestionamentos (CARVALHO et al., 2010).



Figura 1 – Imagem do trânsito (DANTAS, 2011)

Porém, baseado em [Gupte e Younis \(2012\)](#), é possível gerir o tráfego de forma eficiente utilizando redes de veículos ad-hoc (VANET) de maneira eficiente e autonoma.

Segundo [Cavalcanti \(2008\)](#), a crescente quantidade de dispositivos eletrônicos que podem ser embarcados em veículos automotores como DVD, TV, GPS e telefone celular, faz com que os veículos deixem de ser apenas um meio de transporte e passem a oferecer uma rede de serviços e entretenimento para os condutores e/ou passageiros.

## 1.1 Justificativa, Problemática e Hipótese

### 1.1.1 Justificativa

É crescente o número de pesquisas sobre VANETs com intuito de aumentar a segurança viária, bem como ofertar e comercializar novos serviços a motoristas e passageiros. VANETs que usam veículos como nós móveis, são uma subclasse de rede móveis ad hoc chamadas de MANETs (*Mobile Ad hoc Network*), elas fornecem comunicação entre os

veículos próximos e entre veículos e equipamentos à beira da rodovia mas, aparentemente difere de outras redes por suas próprias características do ambiente.

Diante da evolução das redes veiculares e da necessidade de garantir o tráfego de dados dentro da alta mobilidade dos nós numa VANET, constata-se a importância de utilizar tecnologias que permitam auxiliar na solução dos principais desafios relacionados a este tipo de rede.

O uso de VANETs representa uma boa oportunidade para melhora da segurança, eficiência e conforto relacionados ao trânsito nas grandes cidades, porém, possui características peculiares que apresentam obstáculos a serem contornados.

Os nós numa rede VANET são muito dinâmicos pois os veículos possuem velocidade e direção variável. A alta mobilidade dos nós conduz a uma topologia de rede dinâmica caracterizada pela constante perda de comunicação fazendo com que os algoritmos de roteamentos tornem-se complexos e limitados. Normalmente, VANETs apresentam três aspectos para pesquisas: roteamento, segurança e privacidade, bem como suas aplicações (LIANG et al., 2015).

O roteamento em VANETs é baseado na comunicação sem fio e devido à topologia dinâmica e conectividade intermitente, a maioria dos algoritmos em MANETs não estão disponíveis para os vários cenários VANETs, tornando o tema ainda em aberto e sendo foco de várias pesquisas na atualidade com objetivo de dar confiabilidade na comunicação levando em consideração número de emissores e receptores e tipos de roteamento.

Comentado por Cavalcanti (2008), os nós de uma rede veicular apresentam como principal característica, a alta mobilidade, por serem dinâmicos e apresentarem rápidas variações das condições do meio sem fio são os principais geradores de desafios. O alto dinamismo representa um contratempo para as comunicações, visto que nem sempre o tempo de contato dos veículos é suficiente para estabelecer uma comunicação efetiva.

Trabalhos como Hadaller et al. (2007), Nandan et al. (2005) e Chen et al. (2006), apresentam propostas que vão desde a implementação de uma infra-estrutura fixa até definir esquemas de agrupamento de nós, visando reduzir a latência de entrega de mensagens para sistemas de segurança. Nandan et al. (2005) propuseram uma comunicação eficiente para sistemas P2P com disponibilização de conteúdo utilizando um mecanismo de partição dos arquivos disponíveis em pequenos pedaços. Este particionamento visa promover uma forma de transferência distribuída, na qual é possível que um nó esteja recebendo dois pedaços distintos do mesmo arquivo de duas fontes diferentes e simultaneamente. Cabe



ressaltar que, em grande parte, as propostas visam agilizar o processo de transferência de dados em VANETs e implementam mecanismos específicos para minimizar os problemas gerados pela alta mobilidade dos nós.

Diante desse contexto, este trabalho apresenta como principal contribuição um modelo de arquitetura de software flexível e extensível, que utiliza rede veicular e computação em nuvem, com a finalidade de proporcionar uma alternativa para os principais desafios relacionados à VANETs tais como:

- **meio físico:** interferências devido a prédios, árvores e outros obstáculos;
- **alta mobilidade:** dificulta a troca de informações mais completas;
- **topologia:** VANET possui uma característica dinâmica, devido à velocidade que os veículos se movimentam;
- **baixa densidade:** quando a densidade de tráfego é baixa e os veículos estão distantes uns dos outros;
- **alta densidade:** muitos veículos em uma pequena área faz com que a quantidade de mensagens trocadas seja um problema;
- **segurança:** como as VANETs suportam aplicações de emergência em tempo real e lidam com informações críticas de segurança no trânsito, estas devem satisfazer os seguintes requisitos de segurança: confidencialidade, integridade, disponibilidade, autenticidade, privacidade e não repudição para prover segurança na comunicação dos dados (SAMARA; AL-SALIH; SURES, 2010) (MATOS et al., 2013).

Segundo Falchetti, Azurdia-Meza e Cespedes (2015), os principais desafios técnicos que são abordados pela maioria dos pesquisadores, podem ser agrupados em seis categorias: mobilidade, latência, confiabilidade, criticidade (prioridade das mensagens), topologia e segurança.

Portanto, pesquisar sobre redes veiculares e computação em nuvem, traz a possibilidade de construção de uma plataforma capaz de criar uma VANET virtualizada, facilitando a comunicação entre os nós virtuais da rede e simplificando a implementação dos algoritmos de roteamento, segurança e aplicações.

Com o uso da computação em nuvem, os autores Liu, Chen e Chakraborty (2015) utilizaram um controlador SDN (*Networking Defined Software*) para enviar mensagens para os veículos a partir de um RSU (Road-side Unit) conectado ao controlador, porém os veículos deveriam pertencer à mesma geolocalização.



Foi criado por HAJJI e BARGAOUI (2010), a plataforma Testbed que serve para testar a implementação real dos roteadores móveis em uma rede veicular virtualizada. A característica principal consiste na sua capacidade em satisfazer ao mesmo tempo as exigências de mobilidade do veículo e o número elástico de roteadores móveis aplicados.

Eltoweissy, Olariu e Younis (2010) pela primeira vez realizaram uma mudança de paradigma, imaginando um ambiente de VANET baseado em nuvem. Eles vieram com uma nova noção de VANET chamado AVC (*Autónomas Vehiculares Nuvens*). Yan et al. (2013) delinearam os desafios de segurança e privacidade nas nuvens veiculares. No entanto, Olariu et al. discutiram sobre nuvens veiculares de maneira abstrata, sem mencionar uma arquitetura em particular.

Recentemente, Hussain et al. (2012) propuseram três cenários de arquitetura para VANETs baseados em nuvem ou seja VC (*Vehicular Clouds*), VUC (*VANET using Clouds*), e HVC (*Hybrid Vehicular Clouds*) e propôs um esquema conhecido como TIaaS (*Information of traffic as Service*), onde a infraestrutura de nuvem atua para fornecer informações de tráfego para os veículos que se deslocam sobre a estrada.



Qin, Huang e Zhang (2012) propuseram um esquema de roteamento, para VANETs, baseado em nuvem chamado VehiCloud que fornece serviços de roteamento através da infraestrutura de nuvem. Veículos compartilham a sua atual e futuras informações de localização na forma de *waypoints* com infraestrutura de nuvem e, em seguida, a nuvem lhes proporciona uma melhor informação de roteamento.

De acordo com Falchetti, Azurdia-Meza e Cespedes (2015), a integração de VANETs com a nuvem aumenta a utilização das capacidades computacionais que são subutilizados por aplicações de segurança e supera os problemas de roteamento em comunicação V2V (*Vehicle to Vehicle*).

Lee et al. (2014) visam interligar recursos OBU (*On Board Unit*) e RSU (*Road Side Unit*) em nuvem para tarefas cooperativas sensoriais, de armazenamento e de computação, enquanto outros (HUSSAIN et al., 2012) (MERSHAD; ARTAIL, 2013) propõem que as RSUs ajam como gateways para as OBUs para acesso às nuvens tradicionais.

Foi proposto por Lee et al. (2014), o VCN (*Vehicular Cloud Networking*) que está sendo visto como uma revolução para modernizar a tradicional VANET, que integra informações de redes e *Cloud Computing* com VANETs tradicionais. No VCN, veículos e RSU compartilham seus recursos em uma plataforma virtual como ilustrado na figura 2.

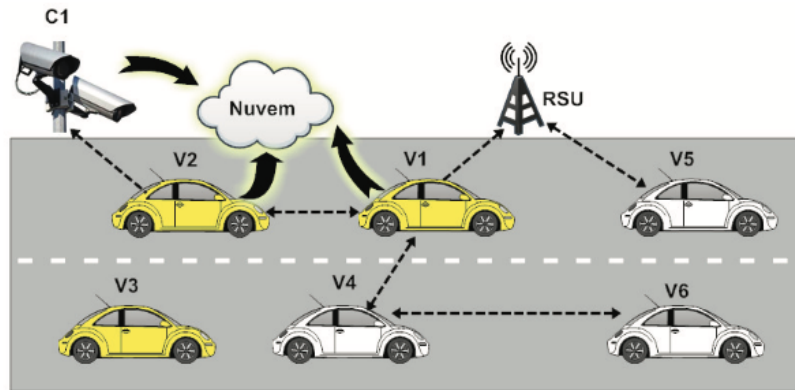


Figura 2 – Modelo do VNC (LEE et al., 2014)

### 1.1.2 Problemática

Como criar uma plataforma em nuvem capaz de permitir a construção de aplicações voltadas para VANETs, considerando o desenvolvimento simplificado e extensível de maneira que o cerne principal da plataforma seja abstraído das novas funcionalidades, mantendo o foco nos principais desafios atualmente encontrados em redes veiculares e permita a escalabilidade.

### 1.1.3 Hipótese

Avaliar se o tempo de resposta da arquitetura proposta permite os mais variados tipos de aplicações em VANETs, dada a particularidade deste tipo de rede.

## 1.2 Objetivos da Dissertação

O objetivo principal desta dissertação é apresentar uma arquitetura de software flexível e extensível, com capacidade de virtualizar nós de uma rede VANET, realizando a comunicação entre os elementos de forma virtual na tentativa de corroborar com a solução de alguns dos principais desafios relacionados às redes veiculares.

### 1.2.1 Objetivos Específicos



Para alcançar a integração anterior, alguns objetivos específicos devem ser atendidos, a saber:

- **Projeta** uma arquitetura de software de maneira que permita a extensibilidade, flexibilidade e escalabilidade;
- **Construir a arquitetura utilizando a linguagem Java;**
- **Implementar uma versão para cada camada da arquitetura;**
- Realizar testes simulados para avaliar seu desempenho e capacidade.

### 1.2.2 Metodologia

No desenvolvimento desta dissertação, foram empregadas a pesquisa bibliográfica, pesquisa quantitativa e a pesquisa experimental.

Na pesquisa bibliográfica foi verificado o estado da arte sobre redes veiculares e seus principais desafios. A pesquisa quantitativa foi utilizada para poder quantificar dentre as aplicações relacionadas à VANETs quais os tempos de comunicação necessários para viabilidade da solução. E por fim, a pesquisa experimental, pois foram realizados experimentos na arquitetura e gerados dados em um ambiente simulado para responder a hipótese levantada na problemática.

Na definição do escopo e das metas desta pesquisa, ficou estabelecido definir e construir uma arquitetura de software modular de maneira que cada módulo abstraia detalhes de implementação dos demais, permitindo usufruir de “contratos” no uso efetivo dos módulos.



Os cenários para os experimentos foram definidos utilizando movimentações reais dentro da cidade de **Aracaju-Sergipe Brasil**, os quais, foram extraídos 12 **milhoes** de movimentações de um sistema de monitoramento de veículos de uma empresa de **taxi**.

Para alcançar os resultados, foi necessário estabelecer conexões entre os nós de uma rede VANETs, permitindo a criação de rede lógica, de maneira dinâmica, “aleatória” e mantendo a integridade dos dados.

A métrica e a técnica de avaliação desta pesquisa foi o tempo de processamento de cada mensagem enviada ao servidor como também, o tempo de comunicação entre os

veículos e os servidores em nuvem, contrapondo entre as velocidades do 2G, 3G, 4G e 5G, permitindo avaliar se as velocidades atuais da telefonia móvel suportam a solução e um teste de carga para medir a capacidade operacional da plataforma.

### 1.2.3 Contribuições

A computação veicular em nuvem é um novo campo de pesquisa com o potencial de mudar a vida das pessoas. Ele traz a eficiência, segurança e conforto para motoristas e passageiros (FALCHETTI; AZURDIA-MEZA; CESPEDES, 2015). Sendo a principal contribuição deste trabalho, a construção de uma arquitetura para criação de redes veiculares em nuvem, como também, a análise dos dados coletados a partir dos experimentos. Nesse sentido, este estudo mostra que também é válido o uso de VCN no auxílio das soluções para os principais desafios relacionados à VANETs.

Outra contribuição importante foi poder instalar e analisar aspectos de segurança nas redes veiculares, podendo apenas assinar a mensagem ou criptografá-la, seguindo com análise de viabilidade de ambos os casos. Conclui-se que, o desempenho e a eficiência da arquitetura atendem aos requisitos relacionados à VANETs. E assim, demonstrar que o tempo de conexão e processamento em uma rede VCN, em ambiente de simulação, atendem às necessidades de vários tipos de aplicações relacionadas a VANETs.

Dentro desse contexto, este trabalho apresenta como principal contribuição, um modelo de arquitetura de software flexível e extensível, que utiliza rede veicular ad-hoc (VANET) e Computação em Nuvem, com a finalidade de proporcionar uma alternativa para os principais desafios relacionados a VANETs .

## 1.3 Organização do Trabalho

Os demais capítulos da dissertação estão organizados da seguinte forma: O capítulo 2 aborda de maneira geral, conceitos de internet das coisas e mostrando a sua importância para soluções de ITS. No capítulo 3, é mostrado **definições** de sistemas distribuídos bem como os tipos e os protocolos de comunicação. É apresentado no capítulo 4, as redes veiculares, seus os principais desafios, seus protocolos e suas aplicações. No capítulo 5 é apresentado a arquitetura I9Vanet, seus módulos e detalhes da tecnologia utilizada na construção. Já o capítulo 6, apresenta o processo de avaliação realizado na arquitetura I9Vanet, bem como,

a análise dos resultados. O capítulo 7 apresenta as conclusões, destacando as principais contribuições, o resultado da análise dos dados e sugestão para os possíveis trabalhos futuros.

## 2 Rede de Sensores sem Fio

As **Redes de Sensores** ou WSN (*Wireless Sensor Network*) estão emergindo como uma nova ferramenta para aplicações importantes em diversos campos como vigilância militar, monitoramento de ambiente, coleta de informações em ambiente hostil, vigilância de edifícios, entre outros.

### 2.1 Definição

Uma rede de sensores sem fio é uma tecnologia específica que ajuda na criação de aplicações com foco em cidades inteligentes. O seu objetivo consiste em criar uma rede que contenha muitos nós de sensores “inteligentes” que possam detectar múltiplos parâmetros de interesse para uma melhor gestão da cidade.

Para realizar tarefas de detecção, a maioria dos Sistemas de Transporte Inteligentes atuais contam com sensores caros, que oferecem apenas funcionalidade limitada. Uma tendência mais recente, consiste em utilizar WSN para tal finalidade, o que reduz o investimento necessário e permite o desenvolvimento de novas aplicações colaborativas e inteligentes, que contribuem ainda mais para melhorar tanto a segurança de condução como a eficiência do tráfego (LOSILLA et al., 2011).

Da mesma forma que em redes de sensores sem fio, os nós de uma rede VANET necessitam trocar informações com outros nós próximos, o que tornam as técnicas de WSN aplicáveis às redes veiculares. Entretanto, há o desafio da alta mobilidade dos nós em uma rede VANET.

### 2.2 Arquitetura de Rede e Topologia

Uma aplicação ITS baseada em WSN distribuída, realiza quatro tarefas principais distintas: (i) aquisição de informação, (ii) distribuição de dados, (iii) processamento de dados para planejar as ações necessárias e, finalmente, (iv) execução das ações apropriadas (LOSILLA et al., 2011)■. Uma vez que estas tarefas possam ser realizadas de forma independente, pode-se considerar que definem correspondentemente quatro subsistemas diferenciados que estão presentes nos sistemas ITS, como mostrado na **figura 3**. Nomeada-

mente o subsistema «Sensores», o subsistema «Distribuição», o subsistema «Tomada de decisões» e o subsistema «Execução».

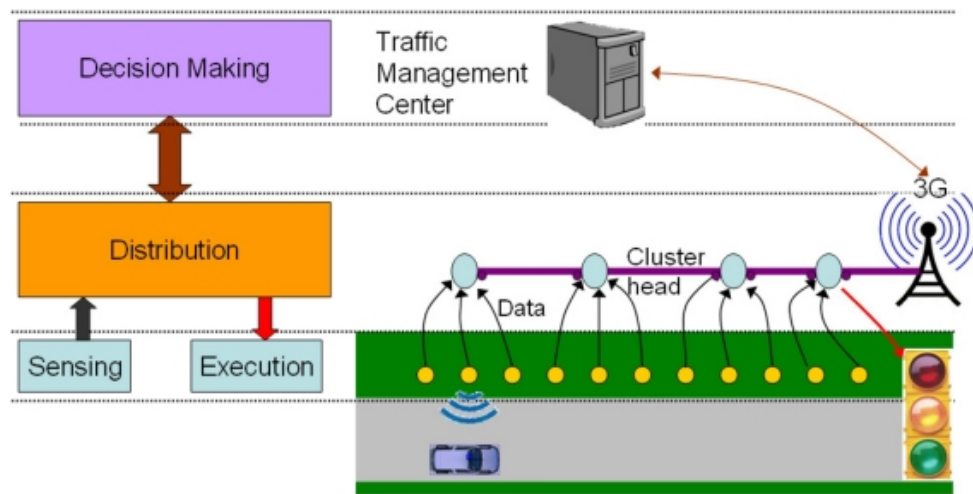


Figura 3 – Arquitetura de referência para aplicações ITS baseadas em WSN. (LOSILLA et al., 2011)

### 2.2.1 Subsistema de Sensores

Este subsistema é composto pela aquisição das informações relevantes, principalmente em relação ao tráfego e estados da rodovia. Numa aplicação ITS baseada em WSN, não necessariamente todos os dispositivos utilizam a tecnologia WSN, permitindo uma distribuição de tarefas entre dispositivos que utilizam tecnologias diferentes.

A implantação do subsistema de sensores consiste na construção de um ou mais WSNs em toda a área de observação (estradas ou parques de estacionamento), que detectam veículos através dos seus sensores e, opcionalmente, comunicam-se por meio da tecnologia sem fio.

Os nós WSN são divididos em grupos seguindo um esquema semelhante, e a implantação consiste em uma composição, tipicamente homogênea, desses grupos de nós. Portanto, eles podem ser considerados como os blocos de construção do subsistema de detecção. Deve-se notar que, além disso, os nós que formam esses blocos podem propagar informações dentro deles, mas isso não deve ser confundido com o subsistema de distribuição. A propagação de dados no subsistema de sensores é restrita a áreas locais, visando extrair informações do bloco / grupo (para um nó coletor) ou ativar o processamento colaborativo



com nós próximos. Se for preciso a disseminação de dados através de áreas maiores, será necessário o uso do subsistema de distribuição.

### 2.2.2 Subsistema de Distribuição

O subsistema de distribuição é responsável pela troca de informações entre os diferentes subsistemas de uma aplicação ITS. Numa arquitetura em camadas, tal como a da Figura 3, ela é colocada numa posição central, recebendo pedidos de comunicação de todos os outros subsistemas e servindo-os em conformidade. Também é responsável pela transmissão dos dados detectados para o subsistema «tomada de decisões» e, por outro lado, pela transmissão de comandos do subsistema «tomada de decisões» para os subsistemas «detecção e execução» (SUNG; YOO; KIM, 2007). Da mesma forma, interconecta os diferentes grupos de sensores de uma rede. Isso resulta em uma rede escalável criada pela composição desses grupos. A este respeito, o subsistema de distribuição pode igualmente interligar grupos fisicamente isolados de nós ou veículos, permitindo assim a interconexão de ilhas WSN desdobradas em diferentes partes ao longo da estrada (WEINGÄRTNER; KARGL, 2007) ou de agrupamentos de veículos que de outra maneira formariam VANETs não ligados (BARBA; AGUIRRE; IGARTUA, 2010) .

O subsistema de distribuição consome uma quantidade significativa de energia devido às exigências impostas aos seus dispositivos. Eles são encarregados de encaminhar cada evento relatado por cada nó do subsistema de sensores, que pode ocorrer em uma alta taxa de frequência. Além disso, deve ser feito sob as restrições de tempo definidas pela aplicação. Isto requer nós muito ativos para garantir a entrega de informações em tempo hábil, comprometendo a economia de energia. Portanto, os dispositivos usados neste subsistema precisam de um orçamento de energia mais generoso do que os nós de sensores, tornando necessárias fontes de energia adicionais (LOSILLA et al., 2011).

Existem diferentes formas de implementar o subsistema de distribuição. Uma delas consiste em empregar redes veiculares para disseminar informações. Dispositivos utilizados nos veículos não têm restrições de energia, uma vez que podem ser alimentados por instalações do próprio veículo. Além disso, a mobilidade de veículos, que é um fator limitante para outros tipos de aplicações, ajuda a espalhar dados em veículos e nós estáticos ao longo da rodovia, e alivia os nós estáticos das tarefas de distribuição. O esquema mais simples envolve a utilização de transmissão direta entre veículos, isto é,

uma rede de um salto que permite a partilha de dados de um veículo de origem para cada veículo que se aproxima (LI; LIU; TANG, 2007) (MIURA; ZHAN; KURODA, 2006). A segunda alternativa baseia-se na utilização de uma rede de distribuição VANET multi-salto (QIN et al., 2010). Esta opção requer mais recursos do sistema, mas também facilita a disseminação mais rápida de dados e escalas muito melhor à medida que o número de veículos tecnologicamente equipados cresce.

De acordo com Losilla et al. (2011), um subsistema de distribuição baseado em redes celulares é uma opção atraente, não apenas por sua alta disponibilidade, mas também pelo custo de implantação. Mesmo sabendo que *gateways* são dispositivos caros que requerem interfaces de rede celular, apenas alguns deles são necessários para serem colocados perto de estações de base celulares (BS) e alguns nós mais baratos que conectam os nós de detecção com os *gateways* devem ser implantados. Por outro lado, os custos de exploração também devem ser levados em conta, uma vez que os operadores de celulares cobram a utilização da BS. Isso pode levar a uma outra escolha, quando não está planejado pagar o uso de BS ou não estão disponíveis, que consiste em usar IMS (Internet Multimedia Subsystems) (BIRK; OSIPOV; ELIASSON, 2009) que fornecem acesso ao Sistema de Distribuição através de diferentes tecnologias (2G, 2.5G, 3G, 4G, 5G, WLAN) independentemente do operador.

### 2.2.3 Subsistema de Tomada de Decisão

Subsistema de tomada de decisão é também conhecido por DMS (*Decision Making subsystem*), é responsável pelo planejamento das ações necessárias para alcançar os objetivos do sistema. As tarefas atribuídas a este subsistema podem ser divididas em três grupos diferentes. O primeiro deles inclui tarefas destinadas ao armazenamento e ao pré-processamento de dados. Trata-se da enorme quantidade de dados que chega ao subsistema, filtrando e armazenando apenas informações relevantes e subseqüentemente, acessando-as. O segundo grupo, trata as informações de tráfego de diferentes fontes e as processa de acordo com o objetivo da aplicação. Finalmente, o terceiro grupo de tarefas é responsável por endereçar comandos de controle, bem como para gerenciamento da rede.

O DMS pode ser executado em diferentes níveis. Em um nível superior, ele pode ser implementado em centrais de monitoramento de tráfego. Isto implica que todos os dados recolhidos pelo subsistema de sensores são enviados, através do subsistema de distribuição, para o DMS, que deve suportar um fluxo de dados assimétrico. A principal vantagem desta

abordagem é a possibilidade de realizar cálculos complexos sobre uma grande quantidade de informação. Inversamente, se apenas o processamento simples for aplicado, o DMS pode ser distribuído entre os nós. Isso permite executar algoritmos colaborativos simples entre nós vizinhos que permitem a execução de aplicações de segurança de tráfego em tempo real. Finalmente, uma outra solução é a utilização de dispositivos inteligentes (smartphones, etc.) em veículos, que podem receber dados brutos de redes rodoviárias e usá-los, por exemplo, para planejar rotas (LOSILLA et al., 2011).

#### 2.2.4 Subsistema de Execução

Este subsistema é responsável por executar ações necessárias que promovam mudanças no fluxo de tráfego de acordo com o objetivo da aplicação ITS. É composto principalmente por dispositivos que fornecem estímulos visuais e acústicos aos condutores, embora outros, destinados à automação de veículos, também pertençam a este subsistema. Os semáforos ou os sinais de mensagem variável instalados ao longo das estradas são opções atrativas que proporcionam um controle rigoroso e adaptabilidade a diferentes situações, respectivamente. Eles oferecem a vantagem de serem infra-estruturas rodoviárias amplamente adotadas, adequadas para reutilização em aplicações para ITS, o que ajudaria a reduzir os custos de implantação (LOSILLA et al., 2011).

O emprego de sistemas no veículo, oferece, por um lado, a possibilidade de apresentar informação personalizada para cada veículo e, por outro lado, a possibilidade de utilizar sinais acústicos e mensagens que diminuem as distrações durante a condução. Além disso, as informações provenientes dos sistemas rodoviários podem ser integradas nos sistemas de informações no veículo, o IVI (*In-Vehicle Infotainment*), por exemplo, para a sua fusão com mapas digitais ou outros serviços de informação (horários de transporte, previsões meteorológicas, etc.). Atualmente, o número considerável e crescente de smartphones, navegadores ou tablets abre o caminho para a adoção de sistemas dentro do veículo. No entanto, há um interesse crescente dos fabricantes de veículos em incorporar sistemas IVI em seus produtos como um diferencial competitivo. A este respeito, novos modelos de automóveis de empresas importantes estão equipados com sistemas como o iDrive da BMW (NIEDERMAIER et al., 2009), o Audi MMI, a Ford SYNC ou GENIVI Apollo da aliança GENIVI apud Losilla et al. (2011), que visam facilitar o desenvolvimento das aplicações IVI.

## 2.3 Considerações Finais do Capítulo



Neste capítulo foi abordado a importância das redes de sensores sem fio em soluções voltadas para cidades inteligentes e principalmente a arquitetura de referência aplicada a ITS. Também foi apresentado a arquitetura de rede e topologia com os subsistemas que o compõe.

## 3 Sistemas Distribuídos

Com o surgimento das redes locais (LAN), o homem tenta aproveitar melhor o poder computacional dividindo tarefas em vários computadores para realizar processamento paralelo, de modo a aumentar o poder computacional sem utilizar super computadores. Surgindo assim, o conceito de sistemas distribuídos (SD).

### 3.1 Definição

De acordo com [Tanenbaum e Steen \(2007\)](#), um sistema distribuído é um conjunto de computadores independentes entre si, que se apresenta a seus usuários como um sistema único e coerente. Já [Coulouris et al. \(2013\)](#), define sistemas distribuídos como sendo uma coleção de computadores autônomos interligados através de uma rede de computadores e equipados com software que permita o compartilhamento de hardware, software e dados.

De acordo com as definições, os sistemas distribuídos apresentam algumas características: concorrência de recursos, falta de um relógio e falhas de componentes independentes. Sendo o compartilhamento de recursos um dos principais motivos para se utilizar sistemas distribuídos.

Os desafios atrelados à construção de sistemas distribuídos são: a heterogeneidade de seus componentes, ser um sistema aberto, segurança e escalabilidade, tolerância a falhas, a concorrência aos recursos e a transparência (COULOURIS et al., 2013).

A miniaturização dos dispositivos e a interligação em redes sem fio, tem ocasionado uma convergência de equipamentos cada vez menores com sistemas distribuídos. A portabilidade desses dispositivos, torna a computação móvel possível e fundamental para que a computação ubíqua ou pervasiva seja utilizada cada vez mais pelo usuário e de maneira transparente e natural.

### 3.2 Características e Desafios

Os sistemas distribuídos são encontrados em toda parte, contudo, há desafios que todo sistema SD precisa se preocupar tais como: heterogeneidade, sistemas abertos, segurança, escalabilidade e tolerância a falhas.

### 3.2.1 Heterogeneidade

Segundo Coulouris et al. (2013), os aspectos que devem ser levados em consideração pela heterogeneidade, estão relacionados à rede, hardware, sistemas operacionais, linguagens de programação e implementações.

### 3.2.2 Sistemas Abertos

Um sistema é considerado aberto quando é possível estendê-lo de várias maneiras. O fato de um SD ser ou não um sistema aberto é determinado pelo grau com que os novos serviços podem ser inseridos para uso por uma variedade de aplicações clientes (COULOURIS et al., 2013).

Para que um sistema seja considerado aberto, é necessário publicar as principais interfaces dos componentes de software. Entretanto, a publicação do padrão de comunicação com o software é apenas o início para adicionar novos serviços a um sistema distribuído.

### 3.2.3 Segurança

A segurança atrelada a sistemas distribuídos possui três componentes que são eles: confidencialidade, integridade e disponibilidade. O primeiro item, deve proteger o acesso à informação de pessoas não autorizadas. Quanto à integridade, os sistemas distribuídos devem garantir a não adulteração dos dados e por último, a disponibilidade, o qual deve proteger contra interferência de acesso ao recurso.

### 3.2.4 Escalabilidade

Em se tratando de escalabilidade, há dois tipos: a vertical e a horizontal, sendo a primeira, a forma de escalonar um sistema centralizado, realizando um *upgrade* na memória, nos processadores e/ou nos discos, quando possível. Já no escalonamento horizontal, consiste em acoplar mais máquinas em um ambiente distribuído, não havendo limites.

Um dos grandes objetivos dos sistemas distribuídos, é permitir o aumento de capacidade à medida que cresce a demanda pelo recurso. E um dos principais desafios é balancear os custos dos recursos físicos com a perda de desempenho, impedindo que

os recursos de software se esgotem evitando gargalos na performance. Sendo assim, a escalabilidade é considerada um tema central em sistemas distribuídos.

### 3.2.5 Tolerância a Falhas

**Infelizmente** os sistemas de computador falham, seja por motivo de hardware ou software, os programas podem produzir resultados incorretos. Em um ambiente com sistema distribuído, alguns componentes podem falhar enquanto outros continuam funcionando, sendo considerado algo complexo tratar corretamente cada problema que pode ocorrer. A tolerância a falhas é fundamental para o bom funcionamento das aplicações de segurança crítica (GORENDER; MACÊDO, 2002) . Dentre as técnicas para tolerância a falhas temos:

- **Detecção de Falhas:** algumas falhas podem ser detectadas antes que mesmo que afete o sistema. Para isso, somas de verificação podem ser utilizadas para verificar a integridade da informação por exemplo, mas outras falhas não há como prevê, como, por exemplo, a parada de um servidor.
- **Mascaramento de Falhas:** algumas falhas podem ser mascaradas, por exemplo, a lentidão momentânea de um *link* pode ser resolvido retransmitindo novamente a mensagem, sem precisar que o usuário tome conhecimento.
- **Recuperação de Falhas:** envolve desenvolver softwares capazes de recuperar ou retroceder o estado dos dados para um momento consistente.
- **Redundância:** alguns serviços podem melhorar sua tolerancia a falhas com o uso de discos espelhados ou *links* de Internet duplicados, entre outras redundancias. Assim, caso algum destes recursos falhem, terá uma alternativa para continuar funcionando.

### 3.2.6 Transparência

A transparência tem como objetivo “esconder” do usuário ou programador detalhes da separação dos componentes em um sistema distribuído, de maneira que seja percebido como um todo.

### 3.3 Comunicação em Sistemas Distribuídos

As restrições temporais determinam a variação dos modelos de sistemas distribuídos existentes. Podendo ser síncronos ou assíncronos. O modelo síncrono deve ser utilizado quando se conhece os limites temporais, permitindo que haja uma estimativa com relação ao tempo de execução dos protocolos do sistema e funções da própria aplicação distribuída (LAMPORT; SHOSTAK; PEASE, 1982) (LYNCH, 1996) (CRISTIAN, 1991). A tolerância a falhas é mais eficiente com a comunicação síncrona, pois permite o uso de *timeout* mais facilmente. Entretanto, nos sistemas assíncronos (*time free*), não há restrições temporais, sendo impossível obter o consenso na presença de falhas (FISCHER; LYNCH; PATERSON, 1985) (MACÊDO, 2000), devido à dificuldade inerente desse tipo de sistema em diferenciar entre uma operação com falha ou extremamente lenta (GORENDER; MACÊDO, 2002).

Os modelos de comunicação existentes para sistemas distribuídos baseiam-se ou nas características síncronas de redes locais ou em ambientes de melhor esforço como os da Internet (GORENDER; MACÊDO, 2002). Nenhum dos modelos consideram as arquiteturas IntServ e DiffServ para prover controle de QoS (Qualidade de serviço).

As arquiteturas IntServ e DiffServ foram propostas pela IETF (*Internet Engineering Task Force*) cujo objetivo é permitir que sistemas de comunicação possam fornecer serviços com diferentes níveis de qualidade, levando em consideração aspectos como: fixação de limite para transferência de dados e diferentes prioridades com relação à possibilidade de perda de pacotes.

#### 3.3.1 Comunicação Cliente-Servidor

A comunicação baseada no Cliente-Servidor foi projetada para suportar a troca de mensagens em interações tipicamente síncronas, pois o processo no cliente bloqueia a execução até a chegada da resposta. Também é considerada confiável, já que a resposta é a confirmação da chegada da requisição no servidor (COULOURIS et al., 2013).

O protocolo utilizado na comunicação Cliente-Servidor é baseado em três primitivas de comunicação: *doOperation*, *getRequest* e *sendReply*. A maioria dos sistemas RMI (Remote Method Invocation) e RPC (*Remote Procedure Call*).

O método *doOperation* é utilizado para invocar processos remotos. Seus parâmetros especificam o objeto remoto e a operação a ser executada. A segunda primitiva, o *getRequest*,



é executado por um processo no servidor, para ler as requisições do serviço. Após execução do processo, pelo servidor, será executado o *sendReply* para enviar a mensagem de resposta para o cliente, fazendo com que o método *doOperation* seja desbloqueado liberando a aplicação cliente.

### 3.3.2 Comunicação em Grupo

Para fornecer tolerância a falhas à disponibilidade, é necessário o uso de difusão seletiva (*multicast*) em situações que precise realizar uma comunicação de um processo com um grupo de processos.

A troca de mensagem *multicast* fornecem uma infra-estrutura importante para construção de sistemas distribuídos com as seguintes características:

- Tolerância a falhas baseada em serviços replicados: as requisições do cliente são solicitadas para todos os membros do grupo onde cada um é responsável por executar a mesma operação. Mesmo se alguma requisição falhar, o cliente pode ser atendido.
- Localizar servidores na interligação em rede espontânea: mensagens *multicast* podem ser utilizadas para localizar os serviços de descobertas disponíveis, podendo ser usadas por servidores e clientes.
- Melhor desempenho através da replicação de dados: em alguns casos, os dados são replicados nas máquinas dos clientes, e quando houver mudança, a atualização é feita por *multicast*.
- Propagação de notificações de um evento: pode ser utilizado para notificar um grupo, de servidores ou clientes, quando ocorrer mudanças em um determinado evento.

### 3.3.3 Protocolos de Comunicação

Há diversos estilos de empacotamento para troca de mensagens em sistemas distribuídos. O CORBA, opta por empacotar os dados para uso pelos destinatários que já conhecem seus tipos anteriormente. Ao contrário do Java, que serializa os dados incluindo informações completas sobre os tipos de seu conteúdo, possibilitando que o destinatário possa reconstruí-los. A linguagem XML segue o modelo semelhante à do Java, fornecendo os tipos de seus conteúdos.

A [figura 4](#) mostra os mecanismos usados para criação de clientes e servidores para *WebService*, CORBA e Java-RMI. Quando são utilizados stubs gerados automaticamente no lado do cliente para Web Services, os processos de desenvolvimento e a complexidade do código para o cliente e do servidor, são praticamente os mesmos para soluções Web Service, Java-RMI e CORBA. Os fatores que determinarão a escolha serão: a interoperabilidade e o desempenho ([GRAY, 2004](#)). Sendo que os Serviços Web permite uma maior integração entre plataformas distintas e dispositivos, enquanto que o desempenho favorecerá ao CORBA e Java-RMI.

[Gray \(2004\)](#) realizou um experimento que enviava uma única solicitação de dados e encerrava. Em qualquer dos três casos, *Web Services*, CORBA e Java-RMI, as tecnologias de serviços da Web funcionam bem em comparação com as outras duas, como mostra a [tabela 1](#).

Tabela 1 – Análise de custos de solicitação de uma única requisição utilizando várias tecnologias([GRAY, 2004](#))

Tecnologia	Latência Total	Total Pacotes	Total Dados Transferidos (B)
WS	0,11s	16	3338
CORBA	0,48s	8	1111
Java-RMI	0,32s	48	7670

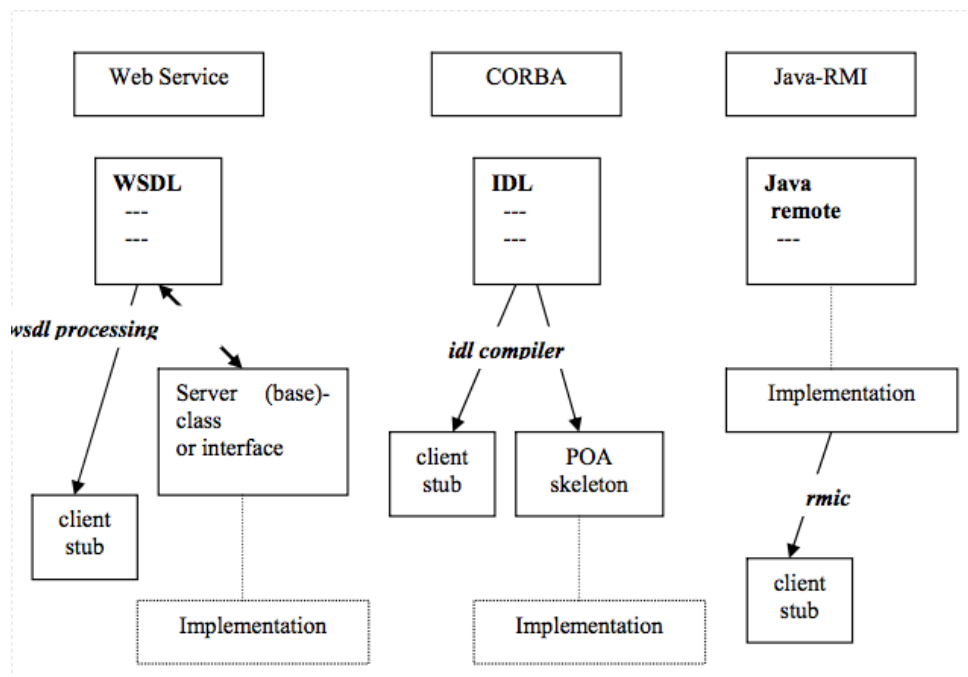


Figura 4 – Geração de componentes de cliente e servidor, a partir da interface para Web Services, CORBA e Java-RMI ([GRAY, 2004](#))

### 3.3.3.1 WebSocket

A tecnologia *websocket* foi introduzida pelo HTML5 (*HyperText Markup Language* quinta geração) através da RFC 6455: *The WebSocket Protocol*, o qual permite utilizar um canal de comunicação bidirecional entre um cliente e um servidor remoto, de forma persistente e dedicado usando um único *socket* TCP. Isto é, a comunicação pode ocorrer nos dois sentidos simultâneo e assíncrono, e a conexão permanece aberta até que em uma das partes realize o fechamento (MELNIKOV; FETTE, 2011).

O *WebSocket* é um protocolo que é definido na camada de aplicação e pode ser utilizado para superar restrições existentes na camada de transporte. Utiliza o modelo de mensagens cliente-servidor como base e suporta tanto dados binários quanto texto simples (THEMUDO, 2014).

As fases de conexão *WebSocet* estão representadas na figura 5, que vai da criação do canal TCP, depois, troca de mensagens e por último, o fechamento. Na fase de **Ligação *WebSocket*** é realizada a troca de mensagens indefinidamente.

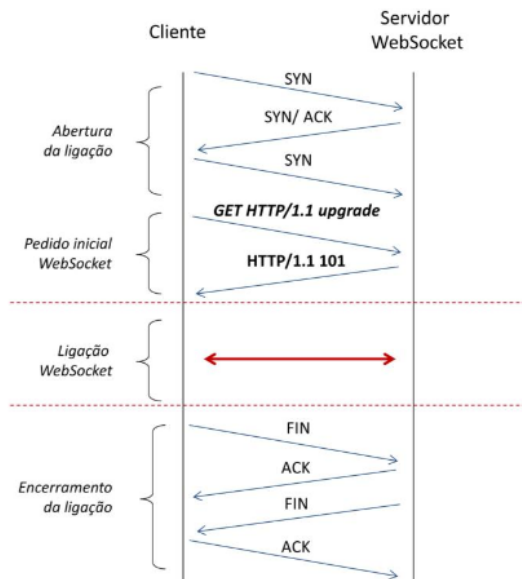


Figura 5 – Etapas de uma conexão *WebSocket* (THEMUDO, 2014)

As implementações do *WebSocket* são baseadas no modelo de eventos, isso quer dizer que as aplicações apenas têm que esperar o evento ser chamado. Há quatro tipos de eventos nas implementações do *WebSocket*:

- **Open:** indica que a conexão foi realizada com o servidor e que a comunicação pode começar.
- **Message:** é acionado sempre que recebe uma nova mensagem.
- **Error:** sempre que ocorre uma falha, normalmente a conexão é finalizada após esse evento.
- **Close:** informa que a conexão foi finalizada.

Sobre a segurança, o *WebSocket* apresenta as mesmas vulnerabilidades em relação a outras aplicações web com o protocolo HTTP (Hypertext Transfer Protocol). Sendo assim, é interessante que a comunicação do protocolo WS seja criptografada com o SSL/TLS. O protocolo SSL/TLS tem como principal objetivo oferecer privacidade e integridade dos dados no momento da comunicação entre dois terminais (THEMUDO, 2014). Na figura 6 percebe-se que o protocolo HTTPS está para o HTTP igual ao WSS com o WS, ambos utilizando o TLS como base.

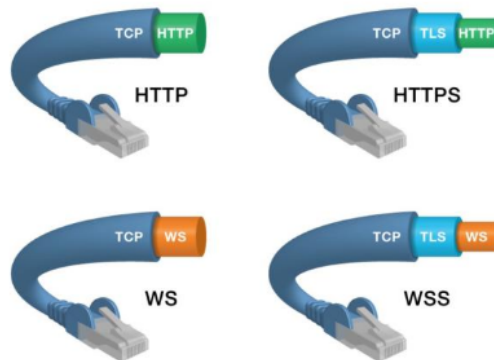


Figura 6 – HTTP, HTTPS, WS e WSS (THEMUDO, 2014)

### 3.4 Considerações Finais do Capítulo

Neste capítulo foi mencionado as características e desafios relacionados à construção de sistemas distribuídos, os tipos de comunicação cliente-servidor e em grupo, como também as tecnologias para se criar sistemas distribuídos tais como: CORBA, JAVA-RMI, WebServices e WebSocket.

## 4 Redes Veiculares

### 4.1 Definição

As VANETs que usam veículos como nós móveis são uma subclasse de rede móveis ad hoc chamadas de MANETs. Elas fornecem comunicação entre os veículos próximos e entre veículos e equipamentos à beira da rodovia. Os nós numa rede VANET são muito mais dinâmicos, pois os veículos possuem velocidade e direção variável. A alta mobilidade dos nós conduz a uma topologia de rede dinâmica caracterizada pela constante perda de comunicação (BUBENIKOVA; DURECH; FRANEKOVA, 2014) (JAKUBIAK; KOUCHERYAVY, 2008) e podem ser categorizadas segundo o tipo de ligações existentes. Dessa maneira, **consideremos** as três arquiteturas de redes veiculares (LUÍS, 2009), como mostra a **figura 7**:



- **Arquitetura WLAN ou celular:** baseada na utilização de antenas fixas, colocadas ao longo da rodovia, funcionando como pontos de acesso à rede. Não existe qualquer ligação direta entre os veículos. Em um cenário de auto-estrada, a implantação dos equipamentos à beira da rodovia de maneira suficiente para permitir a cobertura necessária pode tornar uma solução bastante dispendiosa.
- **Arquitetura ad-hoc:** é considerada uma arquitetura ad-hoc quando não há qualquer uso de infra-estruturas para realizar a comunicação, sendo as ligações é feita diretamente entre os nós envolvidos. Fatores como velocidade ou densidade dos nós podem pôr em *check* o desempenho deste tipo de rede.
- **Arquitetura híbrida:** tem objetivo de retificar as falhas existentes nas duas arquiteturas anteriores, utilizando concomitantemente às arquiteturas ad-hoc e WLAN

As comunicações em VANETs são categorizadas em 4 tipos:

- **Em veículos:** pode ser utilizado para detectar a fadiga e/ou sonolência de um motorista que representa risco na segurança;
- **Entre veículos:** a comunicação V2V (veículo para veículo) pode fornecer uma plataforma de intercâmbio de dados para compartilhamento de informações de advertência de modo a alertar o motorista;

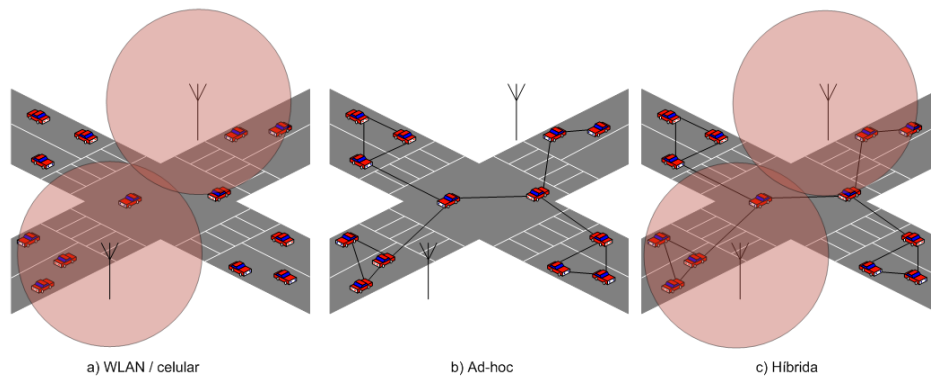


Figura 7 – Arquiteturas de redes veiculares(LUÍS, 2009)

- **Entre veículos e rodovia (V2I):** permite atualização em tempo real do tráfego e fornece detecção e monitoramento do ambiente;
- **Entre veículo e nuvem:** os veículos podem comunicar-se através da banda larga sem fio tais como 3G, 4G ou WIMAX podendo enviar dados para uma central, o que permitiria um controle mais abrangente do tráfego e assistência ao motorista

As aplicações VANETs podem ser divididas em 3 classes: segurança, entretenimento e assistência ao motorista. O principal desafio relacionado à segurança, está ligado à velocidade de alerta ao condutor, para que possa ter tempo hábil de reação. Nas aplicações de entretenimento, destacam-se os sistemas de compartilhamento de conteúdo e jogos. Já sobre a assistência ao motorista, são aquelas que auxiliam o condutor como por exemplo, através do uso de informações sobre as condições do trânsito (SOUZA RONIEL DE; SOARES, 2014).

## 4.2 Características e Desafios

Como mencionado anteriormente, as redes veiculares apresentam como principal característica a sua alta mobilidade, e é justamente por isso que surge uma série de desafios a serem tratados. Dentre os maiores desafios encontram-se:

- **meio físico:** interferências devido a prédios, árvores e outros obstáculos;
- **alta mobilidade:** dificulta a troca de informações mais completas;
- **topologia:** VANET possui uma característica dinâmica, devido à velocidade que os veículos se movimentam;

- **baixa densidade:** quando a densidade de tráfego é baixa e os veículos estão distantes uns dos outros;
- **alta densidade:** muitos veículos em uma pequena área fazem com que a quantidade de mensagens trocadas torne-se um problema;
- **segurança:** como as VANETs suportam aplicações de emergência em tempo real e lidam com informações críticas de segurança no trânsito, estas devem satisfazer os seguintes requisitos de segurança: confidencialidade, integridade, disponibilidade, autenticidade e não repudição para prover segurança na comunicação dos dados (SAMARA; AL-SALIH; SURES, 2010) (MATOS et al., 2013).

Cada um dos desafios relacionados está sendo tratado na arquitetura I9Vanet, uma vez que o objetivo da arquitetura é justamente ajudar a resolver tais problemas.

### 4.3 Segurança

Dito por Wangham et al. (2014), a segurança em redes veiculares é um fator crucial que precisa ser levado em consideração, pois a falta desta pode afetar a vida das pessoas. Como quaisquer redes de computadores sem fio e redes *ad-hoc*, estas estão sensíveis a ataques, tais como: negação de serviço e alteração de mensagens (RAYA; PAPADIMITRATOS; HUBAUX, 2006).

Para construir uma arquitetura de segurança robusta para redes veiculares, é necessário estudar as peculiaridades dos ataques que podem ocorrer. Do mesmo modo que as redes clássicas, as VANETs são vulneráveis a muitos ataques. Alguns destes, são encontrados e soluções são concebidas, considerando que um dia estes ataques podem ser lançados sobre a rede (ENGOULOU et al., 2014). De acordo com Wangham et al. (2014), os principais ataques analisados na literatura são: ataques contra a disponibilidade; ataques contra a autenticidade e a identificação; ataques contra a integridade e confiança dos dados; ataques contra a confidencialidade; entre outros.

#### 4.3.1 Ataques contra a Disponibilidade

- **Negação de serviço (DoS):** este ataque tem como objetivo evitar que veículos autênticos acessem aos recursos da rede não permitindo a troca de informação. Um

ataque pode proceder de 3 maneiras: sobrecarregando um nó específico da rede com informações deixando-o extremamente ocupado; atacando o canal de comunicação gerando altas frequências adicionando ruído e impossibilitando a troca de informações entre os veículos; o não repasse de pacotes para outros veículos da rede, fazendo com que a informação não seja propagada para os outros nós.

- **Negação de serviço distribuída (DDoS):** possui o mesmo objetivo do DoS, ou seja, indisponibilidade de recursos, porém o ataque parte de diferentes localizações e em horários distintos como mostra a figura 8, onde veículos (B, C e D) enviam uma grande quantidade de pacotes contra uma unidade RSU, ocasionando sua indisponibilidade.

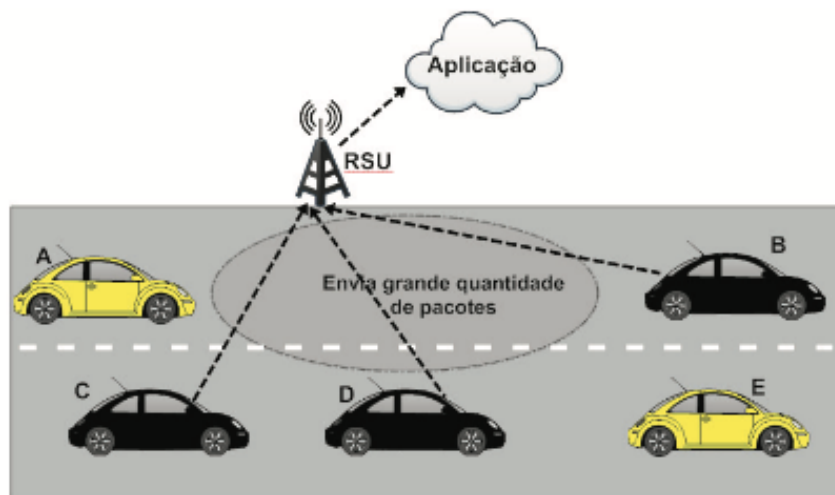


Figura 8 – Ataque DDOS (WANGHAM et al., 2014).

- **Supressão de mensagem:** o atacante recebe e não repassa pacotes da rede com objetivo de impedir que veículos seguintes possam saber de alguma ocorrência, por exemplo, um aviso de congestionamento ou acidentes (TANGADE; MANVI, 2013).
- **Buraco Negro (*black hole*):** é uma área onde o tráfego de rede é redirecionado, contudo, ou não há veículos neste local ou nós maliciosos que estão nesta área se recusam a participar, fazendo com que os pacotes da rede não se propaguem. A figura 9 ilustra o ataque onde veículos se recusam a transmitir a mensagem recebida pelo veículo C.
- **Jamming:** é um ataque de negação de serviço através do meio físico, onde o atacante transmite um sinal para perturbar o canal de comunicação, o que reduz a relação sinal ruído SNR (Signal to Noise Ratio) para o receptor. Segundo Avelar et al. (2015), o impacto com esse tipo de ataque é devastador.



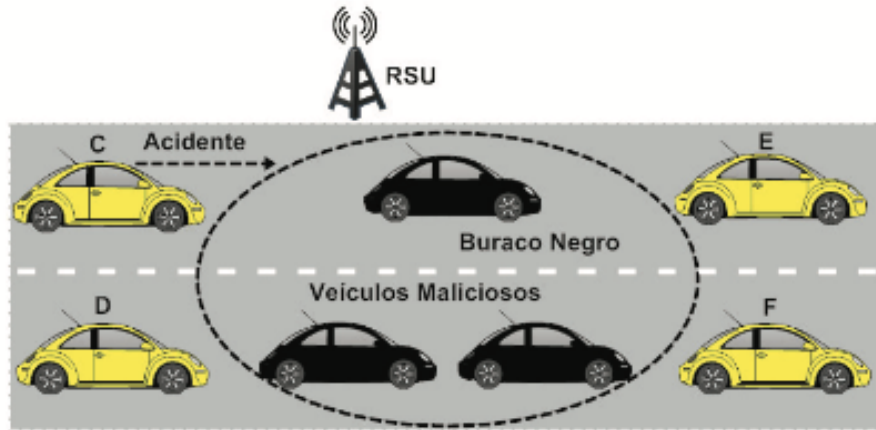


Figura 9 – Ataque Buraco Negro (WANGHAM et al., 2014).

#### 4.3.2 Ataques contra a Autenticidade e a Identificação

- **Falsificação de endereço (*address spoofing*)**: o atacante cria um endereço falso de origem, fazendo com que o nó atacado confie no remetente achando que o mesmo possui permissão para conectar-se à rede (AL-KAHTANI, 2012).
- **Mascaramento**: ocorre quando um veículo malicioso falsifica sua identidade para se passar por outro, com intuito de ter acesso a recursos restritos. Por exemplo, se passar por uma ambulância e ter vantagem no trânsito.
- **Replicação do certificado ou chave**: consiste em utilizar certificados ou chaves duplicados, que são usados como prova de identificação com objetivo de criar ambiguidade dificultando assim a identificação de um veículo pelas autoridades (MEJRI; BEN-OTHTMAN; HAMDI, 2014).
- **Sybil**: é gerado múltiplas identidades por um atacante para simular vários veículos e cada nó transmite mensagens com múltiplas identidades e assim sucessivamente. Desta maneira, um único veículo pode parecer centenas fazendo com que veículos reais mudem a rota achando que ali há um congestionamento (TANGADE; MANVI, 2013).

#### 4.3.3 Ataques contra a Integridade e Confiança dos Dados

- **Falsificação nos dados de GPS (*GPS spoofing*)**: o atacante utiliza um simulador de satélite GPS para gerar sinais mais fortes que o sinal originado de um satélite

real, de maneira a enganar os sensores de GPS, introduzindo uma localização falsa (RAWAT; SHARMA; SUSHIL, 2012).

- **Ilusão (ataque contra os sensores do veículo):** Isaac, Zeadally e Camara (2010) e Al-Kahtani (2012) disseram que esse é um novo tipo de ameaça em aplicações VANETs, onde o atacante interfere intencionalmente nos sensores do seu próprio veículo gerando valores errados, com objetivo de criar mensagens de aviso de tráfego incorretas na rede. Assim, é criada uma condição de ilusão em VANET. Os métodos tradicionais de autenticação e integridade utilizados em redes sem fio são inadequados contra este tipo de ataque.
- **Injeção de informação falsa (*bogus information*):** neste tipo de ataque o atacante pode ser um usuário real ou um intruso que transmite informações falsas na rede para obter vantagens ou afetar decisão de outros veículos. Pode ser chamado também de ataque social, onde o atacante procura confundir e distrair a vítima com envio de mensagens antiéticas, para o motorista ficar confuso e distraído, podendo causar um acidente, como mostra a figura 10.

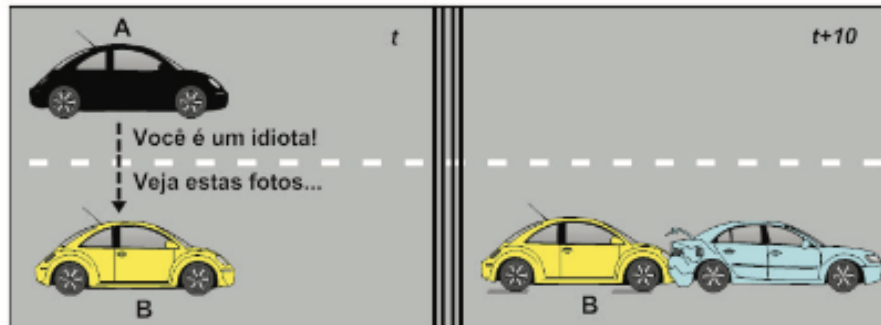


Figura 10 – Ataque social (WANGHAM et al., 2014).

- **Modificação de mensagem (*man in the middle*):** de acordo com Mejri, Ben-Othman e Hamdi (2014), este ataca a autenticidade do remetente e a integridade das mensagens onde o veículo atacante fica inserido entre dois veículos reais que se comunicam. Então, o atacante faz o intermédio entre a comunicação interceptando as mensagens enquanto estes, acreditam estar se comunicando diretamente. A figura 11 mostra o veículo malicioso M escutando a comunicação entre os veículos A e B, modifica um alerta recebido e propaga uma informação falsa, para os veículos B e C )

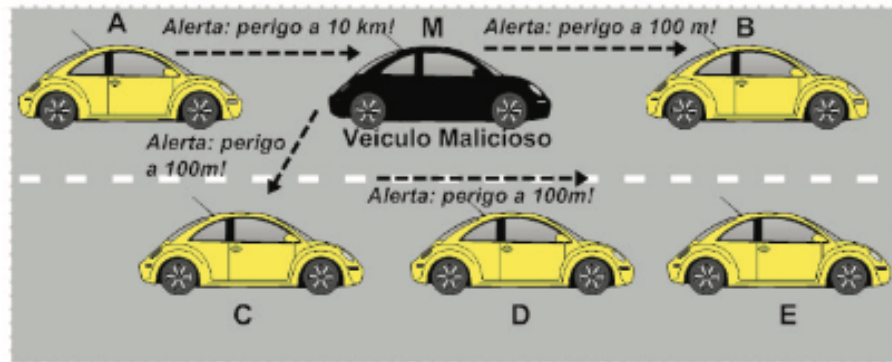


Figura 11 – Ataque demodificação de mensagem (WANGHAM et al., 2014).

#### 4.3.4 Ataques contra a Confidencialidade

- **Análise de tráfego:** segundo Isaac, Zeadally e Camara (2010), esse é um ataque passivo utilizado em redes *ad-hoc*, onde o atacante que tenha acesso à rede pode interceptar o tráfego de pacotes, coletando dados que estejam sendo transmitidos sem o uso de criptografia.
- **Força bruta:** tem como foco capturar mensagens trocadas ou ainda, o processo de identificação e autenticação. Esse ataque não é trivial já que o tempo de comunicação entre os veículos é relativamente curto e esse tipo de ataque consome muito tempo (PATHRE, 2013).
- **Revelação de identidade:** normalmente, o motorista é o dono do veículo, então, o atacante pode obter a identidade do proprietário, violando sua privacidade (TANGADE; MANVI, 2013) e utilizá-lo para compor informações para um ataque social.

#### 4.3.5 Outros Ataques

- **Ataques contra não repúdio (*accountability*):** consiste em tomar medidas para permitir ao atacante, negar a realização de uma ou mais ação (WANGHAM et al., 2014). Apesar disso, dito por Mejri, Ben-Othman e Hamdi (2014), não foi encontrando na literatura um documento afirmando a possibilidade desse ataque.
- **Ataques contra a privacidade:** estes tipos de ataques violam a privacidade dos usuários e condutores em redes veiculares. Segundo Wangham et al. (2014), estudos na literatura classificam esses ataques como uma categoria separada para VANETs e citam dois exemplos: rastreamento de veículo durante sua viagem e engenharia

social, verificando se o veículo encontra-se em deslocamento ou parado (MEJRI; BEN-OTHMAN; HAMD, 2014).

- **Conluio:** é o ataque que se utiliza de vários nós da rede com um objetivo comum, de realizar o vandalismo ou terrorismo, tendo como resultado a indisponibilidade da rede ou de algum serviço ou denegrir a reputação de confiança de um veículo (ZHANG, 2011).

As aplicações suportadas por VANETs, com foco em segurança, permitem a tomada de decisão por um condutor com base em informações enviadas por outros veículos. Entretanto, se um veículo malicioso cria ou altera mensagens, este pode colocar o motorista em situação de risco (WASEF et al., 2010). Dessa forma, a autenticidade das mensagens é fundamental para segurança das redes veiculares e dependendo da aplicabilidade, é necessário ainda fazer uso da criptografia (WANGHAM et al., 2014). Pensando nisso, vários autores (RAYA; PAPADIMITRATOS; HUBAUX, 2006) (MEJRI; BEN-OTHMAN; HAMD, 2014) escolheram a assinatura digital como solução para autenticação das mensagens contidas nas redes veiculares.

O método mais simples e eficiente para assinatura digital, é atribuir, a cada veículo, um par de chaves possibilitando assinar digitalmente as mensagens. Assim, é necessário utilizar uma autoridade certificadora confiável implicando no uso de uma Infraestrutura de Chaves Públicas (ICP) veicular (WASEF et al., 2010).

## 4.4 Algoritmos de Roteamento

Os algoritmos de roteamento em redes veiculares são um dos desafios estabelecidos, já que não existem rotas pré-definidas e nem estimativa da quantidade de nós de uma rede e o fator primordial é a alta mobilidade dos nós. Então, como calcular as rotas necessárias para encaminhar os pacotes com sucesso desde a origem até ao destino?

Foram realizados diversos estudos com a finalidade de comparar o desempenho dos algoritmos aplicados para redes móveis, de modo a corrigir as suas limitações, alguns sofreram adaptações, enquanto outros foram criados (LUÍS, 2009).

Dentre os principais algoritmos de roteamento, com foco nas redes veiculares, temos os protocolos: ad-hoc, baseados em localização, baseado em *clusters*, por *broadcast* e *geocast*.

#### 4.4.1 Protocolos Ad-hoc

As redes ad-hoc são caracterizadas por ser um tipo de rede que não utiliza infra-estruturas e permite a mobilidade dos nós. Ao contrário dos outros protocolos de roteamento infra-estruturados, os quais não apresentam um desempenho aceitável para redes VANETs.

São tradicionalmente divididos em duas grandes categorias: reativos e pró-ativos. Os Reativos caracterizam-se pelo fato de nem sempre possuírem as rotas para todos os nós da rede. Quando um nó precisa de uma rota para um certo destino na rede, inicia o processo de descoberta de rota e é finalizado quando a rota é calculada com êxito ou não exista uma rota disponível. As rotas já descobertas, são utilizadas enquanto o nó de destino permanecer alcançável ou não ser mais necessário. Os protocolos reativos mais conhecidos são AODV (*Ad hoc On Demand Distance Vector*) e o DSR (*Dynamic Source Routing*). Em contra partida, Boukerche (2004), afirmou que o desempenho dos protocolos AODV e DSR é bastante prejudicado pela constante alteração topológica da rede.

Os protocolos **pro**-ativos (*table-driven*) caracterizam-se por ter conhecimento das rotas para todos os nós existentes. Esse protocolo possui a vantagem de ter um reduzido atraso inicial, pois as rotas já estão estabelecidas. Não obstante, a metodologia dos protocolos pró-ativos implica na existência de um tráfego adicional para controle da topologia, uma vez que é preciso manter as rotas existentes sempre atualizadas, consumindo uma maior largura de banda. São exemplos de protocolos pró-ativos: DSDV (*Destination Sequenced Distance Vector*) e o WRP (Wireless Routing Protocol).

Os protocolos PRAODV (*Preemptive AODV*) e PRAODV-M (*Preemptive AODV - Maximum*) foram criados em cima do protocolo AODV, oferecendo um componente baseado na velocidade, na localização e na predição. A diferença é que no PRAODV é estabelecida uma ligação alternativa entre dois nós, antes mesmo da principal expirar, e a PRAODV-M, escolhe uma rota que prevê ficar ativa por mais tempo, ao contrário da AODV, que escolhe a rota mais curta (LUÍS, 2009).

O protocolo OLSR (Optimized Link State Routing), realiza uma otimização da topologia, sendo este, o algoritmo mais utilizado em redes ad-hoc (JACQUET et al., 2001). Possui um modo de funcionamento bastante característico, pois cada nó, seleciona dentre seus vizinhos, uma quantidade de nós suficientes de maneira a cobrir toda a vizinhança a dois saltos do nó, com objetivo de realizar o roteamento e retransmitir as mensagens.

#### 4.4.2 Protocolos Baseados em Localização

Um dos primeiros protocolos baseados em localização foi o GPSR ([Greedy Perimeter Stateless Routing](#)), no qual baseia-se na informação geográfica em relação aos vizinhos. A vantagem deste protocolo está em manter a informação apenas sobre a topologia local, permitindo uma maior escalabilidade e um menor tempo para criação de novas rotas. Entretanto, o desempenho do GPSR é comprometido em ambiente reais, apresentando obstáculos e distribuição aleatória dos veículos.

O protocolo GSR (*Global State Routing*) foi criado com objetivo de corrigir a limitação do GPSR de modo a utilizar o *link-state* de forma que cada nó mantém uma tabela de conectividade, contendo todas as ligações existentes entre os diversos nós da rede, otimizando as decisões a nível de roteamento local. Avaliação feita por [Füßler et al. \(2003\)](#) indica que o GSR apresentou melhor desempenho em relação ao GPSR e melhora no atraso comparado ao DSR, do mesmo modo que uma melhor taxa de sucesso de entrega e menor ocupação da largura de banda se comparado com o AODV ([LI; WANG, 2007](#)).

O protocolo A-STAR é baseado no GSP e GPSR e incorpora um sistema de sensibilização de tráfego (*traffic awareness*) fazendo uso de mapas das estradas ordenados por utilização, de forma a poder definir suas rotas pelas estradas com maior conectividade, na tentativa de aumentar a probabilidade de sucesso na entrega dos pacotes. Devido à sensibilização de tráfego aplicado ao A-STAR, ele apresenta um melhor desempenho, entorno de 40%, na entrega de pacotes em relação ao protocolo GSR.

Foi proposto por [Leontiadis e Mascolo \(2007\)](#) o protocolo de roteamento denominado GeOpps (*Geographical Opportunistic routing for vehicular networks*), ele assume que todos os nós estão munidos com sistemas de posicionamento de maneira a encaminhar um pacote para um nó que está, teoricamente, em melhores condições de poder entregar ao seu destino final. Segundo [Karp e Kung \(2000\)](#), os resultados mostraram que o protocolo GeOpps tem um melhor comportamento do que o GPSR.

#### 4.4.3 Protocolos Baseado em *Clusters*

Os protocolos de roteamento baseados em *Clusters* representam uma rede virtual criada por meio de nós de uma rede física, onde o grupo criado por um conjunto de nós interligados de maneira lógica, tem tendência a alterar rapidamente a sua composição.

Cada *cluster* pode apresentar um nó como líder, denominado *cluster-head*, que é incubido pela coordenação da comunicação dos nós da rede. Os nós de um cluster podem se comunicar diretamente, porém a comunicação extra grupo deve ser feita somente pelo líder. A criação dos *clusters* é de suma importância para aumentar a escalabilidade dos protocolos de roteamento e está na estabilidade a chave para o desempenho destes algoritmos (LUÍS, 2009).

Em redes veiculares, a aleatoriedade da movimentação dos veículos faz com que os protocolos de roteamento baseados em cluster aplicados em redes móveis sejam frustrados, tais como: *Adaptative Clustering* e o MCDS *Minimum Connected Dominating Set* (DAS; BHARGHAVAN, 1997).

Como dito anteriormente, o sucesso de um cluster está na estabilidade, consequentemente, muitas pesquisas são realizadas com objetivo de tornar o grupo o mais estável possível a exemplo do protocolo COIN *Clustering for Open IVC Networks* que utiliza informações sobre mobilidade para formação do *cluster*. Entretanto, são adicionados ao algoritmo, as intenções do motorista do veículo como também a dinâmica veicular. De acordo com Blum, Eskandarian e Hoffman (2003), resultados demonstram que as otimizações feitas melhoram o desempenho do protocolo, identificando um aumento de 192% no tempo médio de vida de um *cluster* e uma redução de 42% no número de alterações dos integrantes do grupo.

O protocolo CBLR (*Cluster-Based Location Routing algorithm*) apresentado por Santos, Edwards e Edwards (2004), utiliza conceitos de *cluster* juntamente com informações de localização. Este protocolo, se comparado ao AODV e DSR, desmostrou um desempenho superior ao atraso *end-to-end* e à taxa de sucesso de entrega.

#### 4.4.4 Protocolos por Broadcast

Os protocolos de roteamento baseados em broadcast consistem em transmitir informações por todos os nós que façam parte da rede. Em VANETs, este tipo de difusão é muito utilizado para compartilhar informações sobre o tráfego, condições da estrada, condição do clima, entre outros (LUÍS, 2009). Assim, fica garantido que todos receberam a informação. Entretanto, o funcionamento destes protocolos não é indicado para redes consideravelmente grandes, podendo gerar um efeito de tempestade de broadcast (broadcast



storm)), aumentando a probabilidade de colisões de pacotes e o consumo da largura de banda, comprometendo o desempenho.

O protocolo BROADCAST (BROADCAST COMMUNICATIONS) apresenta algumas semelhanças com os protocolos baseados em clusters ao dividir a auto-estrada em células e utilizar o conceito de *cluster-head*, aqui chamando de *cell-reflectors* (DURRESI; DURRESI; BAROLLI, 2005). Neste caso, a diferença é que as *cell-reflectors* devem ficar geometricamente no centro da célula. A função do *cell-reflectors* é difundir as informações de emergências entre as células. Contudo, este protocolo funciona apenas para ambiente de auto-estrada.

#### 4.4.5 Protocolos *Geocast*

O protocolo de *Geocast* (*Geocast Routing*) leva em consideração a posição/localização, e seu objetivo é entregar um pacote aos nós que estão em uma determinada região denominada ZOR (*Zone of Relevance*). A implementação deste protocolo deve levar em consideração a integração de um serviço de multidifusão em conjunto com o agrupamento dos nós conforme seu posicionamento geográfico, criando assim as ZORs.

Uma implementação do algoritmo *Geocast* foi utilizado na construção do protocolo *Message Dissemination Process* proposto por Briesemeister, Schafers e Hommel (2000), cujo objetivo é evitar colisão de pacotes e diminuir o número de retransmissões. No momento que um nó recebe um pacote, ele não o encaminha imediatamente esperando um tempo de modo a poder tomar uma decisão referente à retransmissão. O tempo de espera é baseado na distância do nó que lhe enviou o pacote, ou seja, quanto maior a distância menor é o tempo de espera. Quando o período de tempo expira, o pacote somente é retransmitido se a informação não tiver sido recebida novamente. Esse controle no envio dos pacotes faz com que seja menos provável a existência de *broadcast storms* e a propagação de pacotes seja mais eficiente.

Os protocolos DRG (*Distributed Robust Geocast*) e ROVER (*RObust Vehicular Routing*) projetados por Kihl, Sichitiu e Joshi (2008), sendo o DRG um protocolo com foco em grandes cenários, adaptável às constantes mudanças da topologia das redes veiculares e fornece um sistema de encaminhamento rápido e confiável. Em contra-partida, o protocolo ROVER oferece uma difusão *multicast* confiável, tendo como base, um processo de descoberta de rotas dentro da ZOR.



#### 4.4.6 Comparação dos Protocolos de Roteamentos

Ainda não foi encontrada a melhor forma de se criar um protocolo de roteamento para VANETs, porém há autores que consideram um roteamento baseado em cluster mais viável em relação aos outros modelos (LUÍS, 2009). A aplicabilidade do protocolo faz aumentar o número de pesquisas existentes. Sendo que se há alguns protocolos que possuem desempenho favorável em cenários urbanos, isso já não acontece em ambientes de alta mobilidade, e vice-versa. A [tabela 2](#) faz um apanhado geral sobre os protocolos de roteamento e os cenários mais indicados.

Protocolo de Roteamento	Tipo Protocolo	Informação Sobre Posição	Estrutura Hierárquica	Cenário de Mobilidade
AODV	Unicast	Não	Não	—
DSR	Unicast	Não	Não	—
OLSR	Unicast	Não	Não	—
PRAODV-M	Unicast	Seleção de rotas	Não	Auto-estrada
GPSR	Unicast	Encaminhamento de Pacotes	Não	—
GSR	Unicast	Encaminhamento de Pacotes	Não	Urbano
A-STAR	Unicast	Encaminhamento de Pacotes	Não	Urbano
GeOpps	Unicast	Encaminhamento de Pacotes	Não	Urbano
COIN	Unicast	Formação de Cluster	Sim	Auto-estrada
CBLR	Unicast	Encaminhamento de Pacotes	Sim	Circuito circular e quadrangular
BROADCAST	Broadcast	Formação de células	Sim	Auto-estrada
Msg. Diss. Proc.	Geocast	Encaminhamento de Pacotes	Não	Auto-estrada

DRG	Geocast	Encaminhamento de Pacotes	Não	Auto-estrada
ROVER	Geocast	Encaminhamento de Pacotes	Sim	Auto-estrada

Tabela 2 – Protocolos de roteamento aplicados a redes veiculares (LUÍS, 2009)

## 4.5 Aplicações

Há uma série de aplicações que as redes veiculares podem atuar, porém para cada aplicação é necessário conhecer as características que as cercam, tais como: segurança da informação; tempo de resposta; garantia na entrega; entre outros.

Segundo DANTAS (2011), há uma relação entre velocidade dos veículos e a distância entre eles. Esta relação é diretamente proporcional, indicando que quanto maior a velocidade, maior deve ser a distância entre os veículos. Isto ocorre naturalmente, pois o motorista precisa considerar-se seguro no trânsito. Porém, esta condição é dinâmica variando a cada instante, e para agravar, a relação entre velocidade do fluxo e a distância dos veículos, define um fator de realimentação positiva, e sistemas com esta característica tendem à instabilidade, gerando transtornos no trânsito. Então, um sistema capaz de alertar o motorista caso o mesmo esteja a uma distância “não segura” do veículo da frente, pode ser alcançado com sistemas baseados em VANETs.

Outra aplicação, com foco em redes veiculares, pode ser produzida por meio de alertas de colisão em uma via de fluxo intenso. O condutor possui um tempo de percepção mais um tempo de reação e somente após a soma dos tempos poderá executar uma reação. Esta situação mostra a importância de agregar aos veículos uma rede de comunicação no qual os mesmos e seus motoristas poderão trocar informações de ocorrências emergenciais e preventivas no trânsito. Esta situação fica claramente evidenciada na figura 12.

Há também a possibilidade de aplicações voltadas para o entretenimento com o uso de *chats* ou até mesmo propagandas comerciais nas proximidades do veículos.

Por ser uma área crescente, as redes veiculares tem atraído empresas automobilísticas e órgãos controladores a exemplo do órgão americano de pesquisa de inovação tecnológica, RITA (*Research and Innovative Technology Administration*) o qual é coordenado pelo

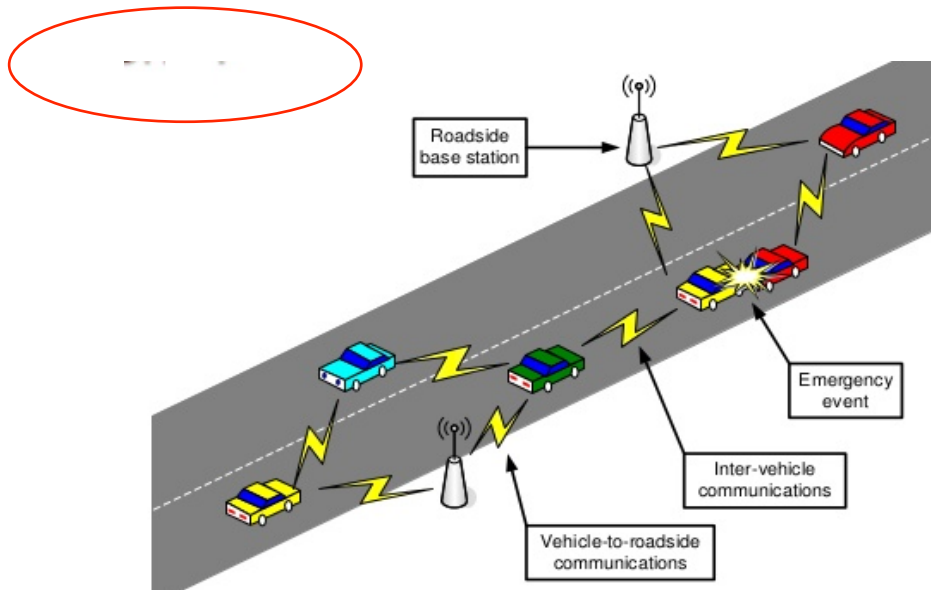


Figura 12 – Exemplo de aplicação VANET (QIAN; LU; MOAYERI, 2008).

departamento de transporte dos EUA. O projeto Car 2 Car (*Communication Consortium*), criado por cinco fabricantes de automóveis europeus (BMW, Volvo, Volkswagen, Honda, e Audi) e apoiada por fornecedores de equipamentos, organizações de pesquisa e outros parceiros, com objetivo de aumentar ainda mais a segurança e a eficiência do tráfego rodoviário através de Sistemas de Transporte Inteligente Cooperativos (C-ITS) com a comunicação veículo para veículo (V2V) suportada pela comunicação veículo infraestrutura (V2I) .

## 4.6 Considerações Finais do Capítulo

Neste capítulo, foi apresentado os principais desafios relacionados à redes veiculares os quais a presente pesquisa se debruça na tentativa de criar alternativas para tais desafios. Explana também sobre as arquiteturas aplicadas às VANETs, a necessidade de segurança e os tipos de ataques possíveis e os principais algoritmos de roteamentos.

## 5 Arquitetura I9Vanet

### 5.1 Visão Geral

A arquitetura proposta tem a finalidade de criar redes veiculares virtuais em nuvem com foco no auxílio das soluções para os principais desafios relacionados à VANETs tais como: alta densidade, baixa densidade, alta mobilidade, segurança e privacidade, entre outros.

A arquitetura consiste em um modelo aberto, dividido em módulos com funções bem definidas. Cada módulo possui funcionalidades específicas e padrões de comportamentos. Sendo possível estender suas operações ou até mesmo substituí-las de maneira que atenda às novas necessidades.

### 5.2 Módulos da Arquitetura I9Vanets

A Figura 13 mostra os módulos necessários da arquitetura I9VANETS: *Applications*, *Server Management Cloud*, *Routing between Nodes*, *Secutity OBU/RSU*, *Infra-Cloud Communication* e *Vehicle-Cloud Communication*. Sendo que cada módulo, segue um modelo de arquitetura com objetivos bem definidos, sendo que suas interfaces de comunicação são padronizadas, permitindo substituir um módulo por outro, com a mesma característica, sem que interfira no modelo.

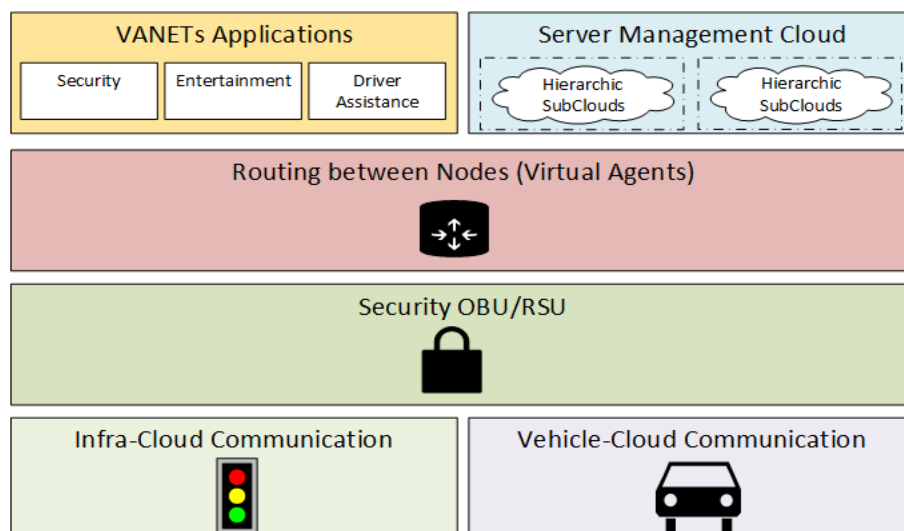


Figura 13 – Módulos definidos para a arquitetura I9Vanet.

Todas as classes que podem ser customizadas, tais como: a tecnologia de comunicação; algoritmo de roteamento; modelo de segurança; implementação de eventos, devem estar definidas no arquivo de configuração **idclassload.properties** como os itens abaixo.

- **ICommunication:**

(br.com.virtualVanets.infraVehicle.communication.CommunicationWebSocketImpl)

Indica a classe responsável por realizar a comunicação dos servidores com os dispositivos.

- **CommunicationV2A:**

(br.com.virtualVanets.infracloud.communication.impl.CommunicationI2AImpl)

Classe que deve enviar e receber os comandos na comunicação entre o Veículo (OBU) e o seu agente virtual.

- **CommunicationI2A:**

(br.com.virtualVanets.infracloud.communication.impl.CommunicationI2AImpl)

Classe que deve enviar e receber os comandos na comunicação entre o dispositivo à beira da rodovia (RSU) e o seu agente virtual.

- **ListenerInfraEquipament:**

(br.com.virtualVanets.applicationModel.listener.DefaultListenerInfraEquipament)

Responsável por tratar os eventos necessários da aplicação com foco na comunicação I2A.

- **ListenerVehicle:**

(br.com.virtualVanets.applicationModel.listener.DefaultListenerVehicle)

Trata os eventos necessários da aplicação com foco na comunicação V2A.

- **EventInfraEquipament:**

(br.com.virtualVanets.infracloud.listener.SVVEventInfraEquipament)

Define a lista dos possíveis eventos de um RSU.

- **EventVehicle:**

(br.com.virtualVanets.vehiclecloud.listener.SVVEventVehicle)

Define a lista dos possíveis eventos de um OBU.

- **SecurityModel:**

(br.com.virtualVanets.security.I9Security)

Responsável por realizar a assinatura e criptografia de todo conteúdo trocado entre os agentes móveis e os dispositivos.

- **RouterNetwork:**

(br.com.virtualVanets.infraVehicle.communication.CommunicationWebSocketImpl)

Implementa o algoritmo de roteamento a ser utilizado na arquitetura.

### 5.2.1 Módulo de Comunicação

O módulo de comunicação é responsável por definir a troca de mensagens entre os servidores e os dispositivos OBU e RSU. Dessa forma, é possível mudar o protocolo de comunicação sem interferir na arquitetura. Basicamente é enviado, dos dispositivos para os servidores em nuvem, informações dos veículos a uma frequência de 10 segundos. Este tempo foi definido devido à alguns modelos de aparelhos de monitoramento GPS veicular utilizarem como menor tempo de envio possível. Exemplos de alguns aparelhos de monitoramento veicular: TK103, AVL05, ST 340, entre outros.

#### 5.2.1.1 Comunicação Infra-Cloud

A forma como a comunicação será efetivamente implementada, não é preocupação da arquitetura, cabe ao desenvolvedor utilizar a tecnologia (socket, webservices, websocket, push, etc) que mais se adequa à sua necessidade e/ou região. As informações trocadas são de entrada e saída, portanto, os equipamentos RSU podem enviar como também receber informações dos servidores. Em VANETs, pode existir a comunicação entre veículos e dispositivos à beira da rodovia (V2I), todavia, não é o propósito desta arquitetura realizar, de forma direta, a comunicação entre os equipamentos, contudo será realizada entre os respectivos agentes virtuais em nuvem.

Todos os equipamentos RSU serão virtualizados e representados por agentes virtuais. Sendo assim, pode haver a comunicação entre agente RSU e o agente do veículo (AV2AI), como também entre agentes RSUs (AI2AI), como mostrado na Figura 14.

Os dispositivos RSUs disponibilizam as seguintes informações: identificação, latitude, longitude, altitude, temperatura, pressão, umidade relativa do ar e um campo textual para informações extras. A plataforma prevê 3 operações:

- **Conectar:** realiza a conexão com o servidor, informando identificador do dispositivo, a latitude e longitude;

- **Enviar Mensagem:** envia mensagens para rede. O conteúdo das mensagens variam de acordo com a aplicação;
- **Desconectar:** fecha a conexão com o servidor.

Para implementar a comunicação entre o agente e o veículo (I2AI) é necessário implementar a classe abstrata **CommunicationI2A** e sobrescrever o método **sendMsg**. Esse método é executado pela arquitetura de maneira transparente e deve enviar o comando para o dispositivo.

#### 5.2.1.2 Comunicação Veículo-Cloud

Define a comunicação entre os servidores e os equipamentos instalados nos veículos (OBU). Os detalhes da implementação podem ser substituídos permitindo utilizar a melhor tecnologia do momento. Assim como no módulo de Comunicação Infra-Cloud, a comunicação V2V não ocorrerá diretamente, como ocorre nos modelos tradicionais, ela será realizada através dos agentes virtuais, que nada mais são do que a representação virtual dos veículos. Desta forma, a comunicação se dará entre os agentes (AV2AV) e entre o agente e o veículo físico (V2A) este processo é mostrado na Figura 14.

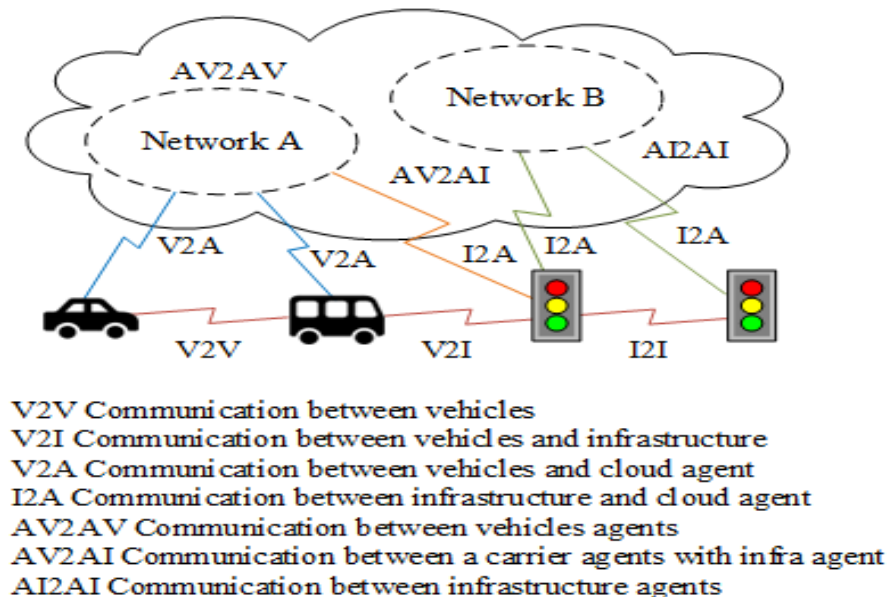


Figura 14 – Modelo de comunicação V2A, I2A, AI2AI e AV2AI.

O veículo deve disponibilizar as seguintes informações: identificação, latitude, longitude, altitude, velocidade, direção, tipo operação e um campo textual. A plataforma prevê 4 operações que podem ser extendidas, são elas:

- **Conectar** (CONNECT\_CODE): realiza a conexão com o servidor, informando identificador do dispositivo, a latitude e longitude;
- **Movimentar** (MOVIMENTATION\_CODE): este comando é executado a cada 10 segundos, com o intuito de informar a geolocalização, velocidade e direção do mesmo;
- **Enviar Mensagem** (SEND\_BROADCAST\_CODE): envia mensagens para os veículos da rede, sendo que o conteúdo das informações variam de acordo com a aplicação;
- **Desconectar** (DISCONNECT\_CODE): fecha a conexão com o servidor removendo-o da rede.

Para implementar a comunicação entre o agente e o veículo (V2AV), é necessário implementar a classe abstrata **CommunicationV2A** e sobrescrever o método **sendMsg**. Este método executa a classe responsável pela conexão efetivamente definida.

### 5.2.2 Módulo de Segurança

A segurança na troca de informações em redes veiculares é um dos desafios que devem ser **solucionado** antes mesmo da implantação de uma solução VANET ser posta **no mercado**. Então, como garantir os princípios da autenticidade, integridade, confidencialidade e não repúdio na troca de mensagens entre os veículos, já que é inviável fornecer as chaves de segurança de todos os veículos durante a comunicação?

Na arquitetura I9Vanet, foi pensado em um processo de comunicação entre os equipamentos, OBU e RSU, e os servidores em nuvem, de forma que deve garantir os princípios da autenticidade, integridade, confidencialidade e não repúdio, através do uso de assinatura digital baseado em algoritmos assimétricos e criptografia dos dados. Toda via, este último foi definido de forma parametrizada para que fosse feito análise comparativa entre a comunicação aberta e criptografada.

A geração das chaves para cada equipamento deve ser controlada pelo órgão responsável pelos veículos, a exemplo do Brasil, o DENATRAN (Departamento Nacional de Trânsito). Quando um novo veículo for cadastrado em sua base, “veículo zero quilômetro”, devem ser geradas as chaves pública e privada, sendo que a pública deve ser enviada para os servidores em nuvem responsáveis pelo gerenciamento da arquitetura I9Vanets e a privada, inserida no equipamento embarcado.



A cada renovação de licenciamento um novo token deve ser gerado e fornecido ao equipamento e publicado nos servidores. Como também, em caso de venda de um veículo, ao passar para o nome do novo proprietário, deve ser gerado um novo token.

Se o parametro de criptografia estiver habilitado, o primeiro passo, no momento da conexão, é a troca de chave secreta para que toda comunicação seja segura. Para isso, assim que o comando *onConnection* for executado, o servidor enviará o comando *createSecretKey* com a chave secreta, criptografada com a chave pública do veículo. Assim o mesmo poderá descriptografar com sua chave privada e utilizá-la na criptografia de cada envio e recebimento de mensagens de agora em diante.

No momento que um equipamento desejar enviar uma mensagem para seu agente em nuvem, ela deve ser assinada com sua chave privada, assim através da verificação da assinatura com a chave pública, o agente poderá confiar no remetente. Porém, a mensagem será **asignada** com a chave privada do servidor para que o equipamento, OBU ou RSU, possa validar através da chave pública do servidor.

Alterar o modelo de segurança adicionando novas regras é possível através da implementação da classe **ASecurityModel** e dos métodos **verifySign**, **sign**, **encrypt** e **decrypt**. A chamada dos métodos é feita pela arquitetura de maneira transparente, não sendo necessário intervenção do usuário.

Como foi dito anteriormente, as redes **veiculares** são vulneráveis a muitos ataques e a arquitetura **procurou** se proteger aos **principais** como mostrado na **tabela 3**.

Tipos de Ataques	Opções	Observação
DoS	X	Este ataque não seria com foco em um OBU ou RSU, apenas fará sentido ocorrendo nos servidores da plataforma I9Vanet. Para proteção os servidores devem se proteger com <i>firewall</i> .
DDoS	X	O mesmo do DoS.
Supressão de mensagem	X	o repasse das mensagens não depende do veículo físico e sim do agente virtual em nuvem.

Buraco Negro	X	quem repassa as mensagens é o agente em nuvem, não sendo possível acontecer este tipo de ataque.
Jamming	–	como a arquitetura depende da por telefonia celular, ou redes wifi, entre outras já mencionadas, então este sinal pode ser afetado impossibilitando o acesso dos veículos à nuvem.
Falsificação de endereço	X	todo nó deve ser reconhecido pelo órgão controlador de veículos por meio de um certificado.
Mascaramento	X	Todo certificado emitido deve ser único e intransferível e renovado anualmente ou a cada operação de compra e venda.
Replicação do certificado	–	Este tipo de ataque envolve proteger o certificado do veículo para que não possa ser utilizado por outro, fugindo do escopo da arquitetura.
Sybil	X	Todo veículo deve possuir um certificado reconhecido por alguma entidade certificadora, não sendo possível simular mais veículos.
Falsificação nos dados de GPS	–	É difícil se proteger deste ataque, pois o sinal de GPS emitido por satélites são abertos.
Ilusão	–	A proteção para esse tipo de ataque consiste em criar uma comunicação inviolável entre os sensores e o equipamento que se conecta na aplicação em nuvem.

Injeção de informação falsa	X	Não há comunicação direta entre os veículos e a aplicação desenvolvida para realizar a comunicação virtual pode realizar uma análise de toda informação.
Modificação de mensagem	X	Não havendo comunicação direta entre os veículos e todas as mensagens sendo assinadas e criptografadas, não há como adulterar uma mensagem.
Análise de tráfego	X	É uma taque em redes ad-hoc, não sendo aplicado à arquitetura.
Força bruta	X	A comunicação é ponto a ponto e o conteúdo é criptografado, dificultando ainda mais este tipo de ataque.
Revelação de identidade	X	A identidade do veículo não é informada durante as comunicações.
Ataques contra não repúdio	X	As mensagens da rede veicular não depende de um veículo para se propagar.
Ataques contra a privacidade	X	Todo conteúdo pode ser criptografado, assim a privacidade do veículo é garantido.
Conluio	X	A segurança é garantida pela arquitetura e não por critérios de confiança que pode ser denegrido por este tipo de ataque.

Tabela 3 – Quadro de análise sobre os tipos de ataques e a arquitetura I9Vanets.

### 5.2.3 Módulo de Gerenciamento dos Servidores

Não é possível prever a quantidade de veículos conectados a um servidor, e isto criou a necessidade de construir uma plataforma de gerenciamento com objetivo de aumentar a capacidade de dispositivos conectados à arquitetura. Então, pensando nisso, foi definido que os servidores devem ser distribuídos hierarquicamente de forma que permitam controlar

regiões, denominadas domínios, separadamente e independentemente. Cada servidor deverá conter informações do **servidor pai**, no caso, o servidor nível 0, o qual não possui pai. A **figura 15** mostra como exemplos de domínios, onde cada cerca virtual representa um servidor.

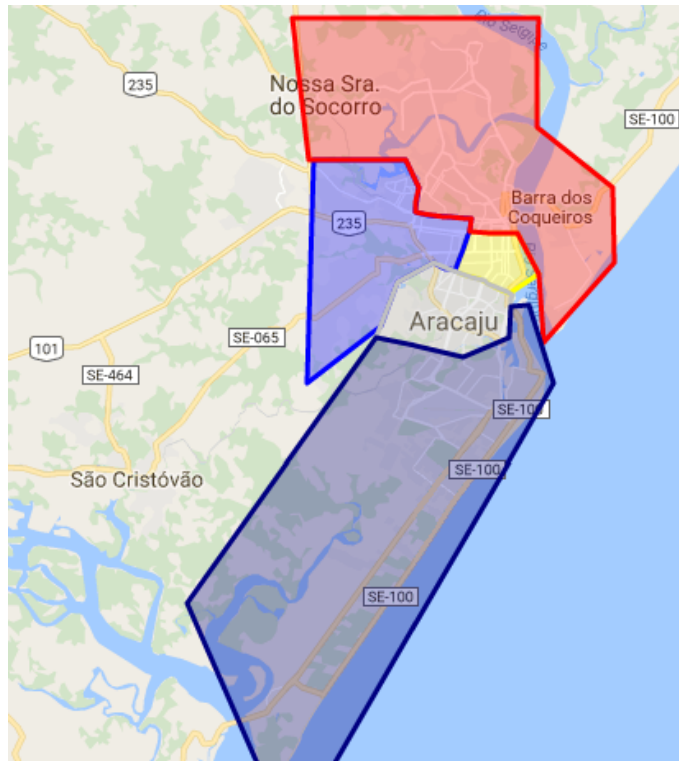


Figura 15 – Exemplo de domínios.

Quando um veículo iniciar a transmissão será conectado automaticamente ao servidor Nível 0, informando o comando *StartConnection*, o qual realizará uma busca pela coordenada GPS enviada pelo veículo para localizar o servidor responsável pelo domínio onde encontra-se o veículo, podendo ocorrer duas situações:

1. O veículo apresenta-se em uma área sem cobertura de cerca virtual, assim o próprio servidor raiz será responsável por gerenciar a rede VANET deste veículo;
2. O veículo encontra-se em uma domínio válido que é de responsabilidade de algum servidor filho. Então o servidor nível 0 irá passar o comando, *ChangeServer*, informando o ip do servidor que deve ser responsável pela rede desse domínio, e assim o veículo faz uma nova conexão para o novo servidor.

Cada movimentação do veículo deve ser transmitida para o servidor do domínio, pelo comando *SendMoviment*, o qual verificará se o veículo ainda está em sua região, de acordo com a cerca virtual. Caso não esteja mais sob seu domínio, o servidor irá enviar o

comando *ChangeServer* com o endereço do servidor **P**ai. Assim, o veículo poderá refazer a conexão com este novo servidor, através do comando *StartConnection*, o qual deverá verificar se a coordenada pertence a este servidor ou a algum de seus filhos. Este processo é realizado para todos os níveis da hierarquia.

Em todos os servidores devem existir uma base de dados para definir a hierarquia, os domínios geográficos como também o cadastro dos dispositivos. As tabelas 4 e 5 mostram, respectivamente, o dicionário de dados da tabela *server* e *device* utilizadas no sistema.

Tabela <i>Server</i>		
Coluna	Tipo do Dado	Descrição
server_id	<i>integer</i>	Identificador único no sistema
address	<i>varchar</i>	Endereço IP do servidor
server_idsuper	<i>integer</i>	Identificador do servidor pai
dominio	<i>polygon</i>	Cerca virtual de responsabilidade do servidor

Tabela 4 – Colunas das tabelas *Server*.

Tabela <i>Device</i>		
Coluna	Tipo do Dado	Descrição
device_id	<i>integer</i>	Identificador único no sistema
identification	<i>varchar</i>	Identificador do dispositivo (Imei, Mac Address, etc.)
type	<i>char</i>	Tipo do dispositivo (OBU ou RSU)
public_key	<i>byte[]</i> <i>integer</i>	Chave pública do dispositivo

Tabela 5 – Colunas da tabela *Device*.

A Figura 16 mostra como os servidores devem ser organizados para melhor controle. O servidor nível 0 é a raiz da árvore e responsável por gerenciar os nós abaixo dele. A estrutura não é um uma árvore binária e sim uma formação em árvore com N nós filhos.

Quando um veículo que está conectado a um servidor precisar conectar-se a outro, a mudança deve ser solicitada ao servidor pai imediato, caso este não seja o de destino, deve perguntar ao pai deste e assim sucessivamente, até encontrar o servidor responsável pelo domínio de destino ou o servidor nível 0 será responsável por este gerenciamento. .

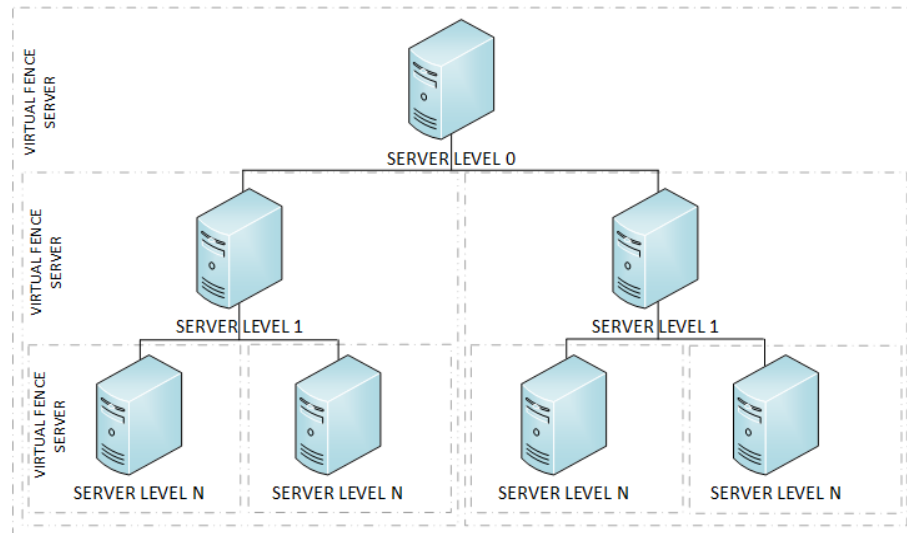


Figura 16 – Organização dos servidores na nuvem.

A cada envio do comando *SendMoviment* pelos veículos, o servidor deve verificar se o mesmo ainda encontra-se no mesmo domínio. Como este processo é executado a cada 10 segundos por cada veículo, foi necessário realizar essa consulta em memória ao invés de ir no banco. Para isso, foi utilizado a API GEOTools.

O GeoTools é uma biblioteca feita em Java *open source* que provê ferramentas para manipulação de dados geoespacial em memória. O objetivo da biblioteca é permitir que programadores que estejam desenvolvendo aplicações geoespaciais se concentrem na construção do negócio fim, enquanto reutilizam ferramentas genéricas para funções básicas (HALL; LEAHY, 2008).

A mesma funcionalidade do GeoTools poderia ser atingida com as funções do **Postgis**, porém ao realizar pequenos testes, foi percebido que o tempo da consulta é 10 vezes maior que a do GeoTools, o que poderia comprometer o processo de avaliação da arquitetura.

### 5.2.4 Módulo de Roteamento

O módulo de roteamento em uma rede veicular é um dos grandes desafios, e por isto é foco de estudos na busca de soluções que viabilizem a rede. Na arquitetura I9Vanet proposta, essa preocupação está presente, porém a plataforma permite criar novos algoritmos sem a necessidade de preocupar-se com detalhes, como segurança, protocolo de comunicação ou quantidade de veículos.

Toda comunicação entre os nós é realizada entre os agentes virtuais da rede (AV2AV e AV2AI e AI2AI). As regras de roteamento podem ser substituídas ou expandidas, basta implementar novos algoritmos seguindo o modelo da arquitetura do módulo. Quando houver a necessidade de enviar algo para algum nó físico da rede, esta mensagem deve ser enviada através do Módulo Comunicação Veículo-Cloud ou Infra-Cloud e eles se responsabilizarão pela entrega.

O módulo de roteamento pode ser definido de acordo com a necessidade, bastando implementar a classe **RouterNetwork**, porém para fins de avaliação da arquitetura foi implementado uma adaptação do algoritmo conhecido como *geocast routing*, no qual o objetivo é entregar um pacote aos nós que pertencam a uma certa região, denominada *Zone of Relevance* que utiliza um OBU como nó cabeça de uma rede veicular.

Os veículos quando iniciam o processo de conexão com o servidor, buscam uma rede já existente para se conectar. O veículo “cabeça” desta rede deve estar a uma distância máxima de 1 KM e seguir na mesma direção na via, caso contrário, este veículo criará uma nova rede esperando que novos veículos solicitem acesso.

A cada transmissão de um veículo, através da operação **sendMoviment**, o algoritmo de roteamento seguirá uma dentre três opções, são elas: permanecer na rede atual; entrar em uma nova; criar uma nova rede veicular.

### 5.2.5 Módulo de Aplicações

Permite adicionar novas aplicações tendo como base os recursos disponibilizados pelos outros módulos. Porém, para que as aplicações possam enviar e receber informações da arquitetura, é necessário definir um contrato com as classes dos módulos, através do uso de interfaces e classes abstratas. Sendo assim, quando um veículo enviar uma informação



para seu agente virtual, a arquitetura irá executar um método padronizado de uma classe da aplicação, fazendo com que os dados sejam entregues corretamente.

Podem ser implementados vários tipos de aplicações com os dados que a arquitetura já disponibiliza, porém caso seja necessário que uma aplicação possa modificar os dados de entrada e saída da arquitetura, é possível utilizar o campo mensagem para adicionar novas informações e assim atender às novas demandas.

### 5.3 Processo de Negócio

A figura 17 e 18 mostram os diagramas de processos de negócio da arquitetura I9Vanet com e sem criptografia, respectivamente. No primeiro processo, o início, figura 17(1), se dá quando o veículo é ligado e abre a primeira conexão com o servidor raiz, figura 17(2). Em seguida, o processo de negociação da chave a ser utilizada na criptografia figura 17(3), a chave é criptografada, pelo servidor com a chave pública do veículo e enviada para o nó da rede, figura 17(4). A partir deste momento, toda comunicação será criptografada e em ambos os modelos, toda mensagem será assinada.

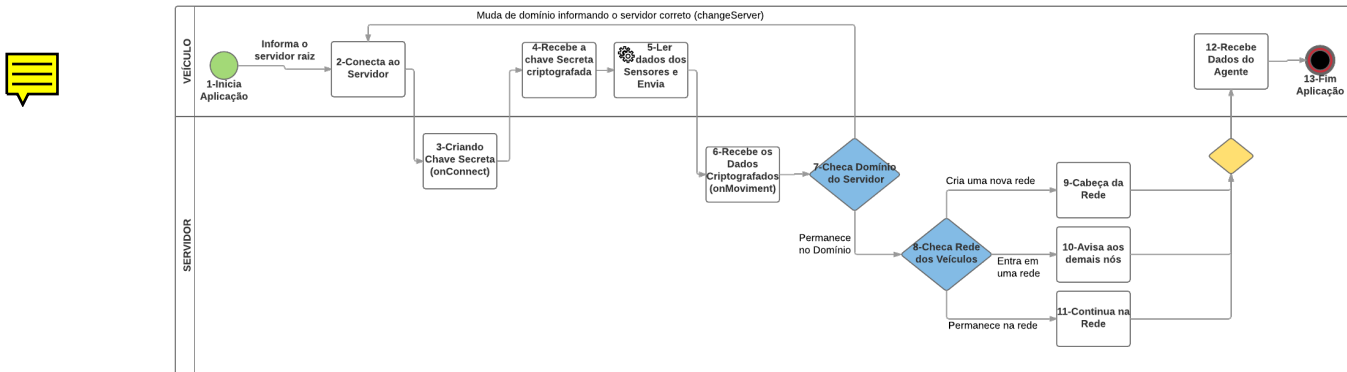


Figura 17 – Modelo de processo de negócio da arquitetura I9Vanet com criptografia.

Após estabelecido o canal seguro, é iniciada uma thread, figura 17(5), responsável por obter os dados dos sensores do veículo e enviá-los para o servidor, figura 17(6), dentre os dados estão a latitude, longitude, velocidade e direção. O recebimento dos dados é feito pelo método **onMoviment** que irá verificar em qual domínio o servidor deve ficar conectado, figura 17(7). Se a coordenada geográfica indicar que o veículo deve mudar de servidor, então, o comando **changeServer** é acionado informando o endereço do servidor “pai”, voltando para o processo, figura 17(2), para recriar a conexão e segurança do canal. Caso permaneça no mesmo domínio, deve realizar uma análise sobre a rede que o veículo



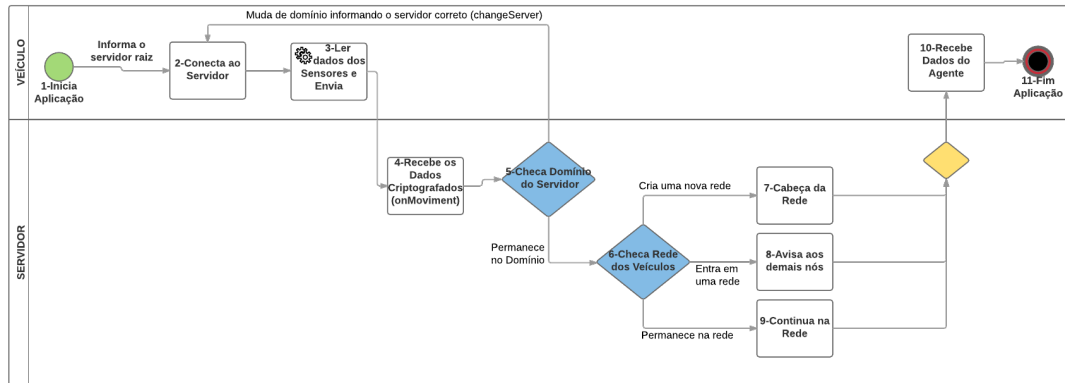


Figura 18 – Modelo de processo de negócio da arquitetura I9Vanet sem criptografia.

pertence, verificando se deve permanecer na rede atual, figura 17(11), cria uma nova rede, figura 17(9), ou entrar em uma outra, 17(10).



O processo de execução da rede em si não está mapeado nesse fluxo, já que as mensagens a serem trocadas entre os nós da rede dependem da camada da aplicação que está sendo desenvolvida.

## 5.4 Considerações Finais do Capítulo

Neste capítulo é proposto a arquitetura I9Vanet e os seus seis módulos: Comunicação Infra-Cloud, Comunicação Veículo-Cloud, Segurança, Gerenciamento de Servidores, roteamento e aplicações. É explicado também, como é feito a comunicação entre módulos e como pode ser realizado uma extensibilidade da arquitetura.

## 6 Avaliação da Arquitetura

Este capítulo visa avaliar a arquitetura I9Vanets por meio do paradigma GQM (Goal-Question-Metric) muito usado para planejar medições em projetos de softwares (SOLINGEN et al., 2002). As seguintes fases foram analisadas durante o processo de avaliação: fase de planejamento, de coleta de dados e de interpretação dos resultados.

O processo de avaliação iniciou-se na fase de planejamento com o intuito de analisar a arquitetura com relação à eficácia e eficiência da solução. O experimento teve como público alvo os desenvolvedores de software interessados na arquitetura como gerenciamento de uma rede VANET para ITS no contexto de cidades inteligentes.

Com base na proposta da arquitetura i9Vanets, foram realizados dois tipos de experimentos. O primeiro, com o intuito de avaliar a sua aplicabilidade em ambientes de redes veiculares reais, teve a função de avaliar o fluxo de dados referentes às velocidades das tecnologias atualmente utilizados na telefonia móvel, sendo 2G, 3G, 4G e 5G. O segundo experimento objetivou dimensionar a capacidade de operação e o comportamento da arquitetura proposta a partir de um excessivas requisições, sem limite de velocidade. Ambos os testes extraíram dados estatísticos a partir de métricas pré-definidas e planejadas.

### 6.0.1 Definição

Analisar a arquitetura I9Vanets com finalidade de avaliar sua eficiência, a respeito do tempo de latência e capacidade de processamento, do ponto de vista de múltiplos acessos concorrentes no contexto de redes veiculares em nuvem.

### 6.0.2 Planejamento

O experimento tem como alvo, os desenvolvedores de soluções que visam melhorar a mobilidade urbana com o uso de VANETs. Onde o objetivo foi avaliar a capacidade de processamento do servidor e medir o tempo de latência das comunicações da telefonia móvel, através dos seguintes questionamentos:

- Qual a taxa de processamento por minuto?
- Qual a latência média de cada requisição para as velocidades de 2G, 3G, 4G e 5G?

- A latência média com a mensagem criptografada irá aumentar em relação à mensagem aberta?
- Qual a capacidade de processamento da arquitetura I9Vanet ?

Para isso, serão utilizadas as seguintes métricas:

- número Total de requisições por min (TR/min);
- tempo de latência da comunicação (Lat);
- tempo de processamento de cada requisição no servidor (PT).

O tempo de latência representa o tempo que uma mensagem sai do ponto A para o ponto B, sendo que o tempo calculado pelo experimento foi o RTT (round trip time), tempo que a mensagem foi enviada e recebida de volta pelo veículo sendo considerado como latência, a metade do RTT.

De acordo com [Papadimitratos et al. \(2008\)](#), as aplicações voltadas para as redes VANETs possuem alguns requisitos que devem ser respeitados e estão relacionados ao tipo de comunicação, ao tipo de mensagem, ao tempo de entrega, à latência (tempo de atraso máximo requerido) e a outros requisitos como mostra a [tabela 6](#). Então, o objetivo do experimento é avaliar se o tempo obtido se adequa aos requisitos mostrados.

Tabela 6 – Características das aplicações veiculares ([PAPADIMITRATOS et al., 2008](#)).

Aplicações	Tempo	Latência	Outros
Alerta de Veículo Lento	500ms	100ms	Alcance: 300m, alta prioridade
Alerta de Colisão em cruzamento	100ms	100ms	Posicionamento preciso em um mapa digital, alta prioridade
Pré Colisão	100ms	50ms	Alcance 50m, prioridade alta/média
Gerenciamento de Cruzamento	1000ms	50ms	Precisão de posicionamento menor que 5m
Download de Mídia	—	500ms	Acesso a internet e Gerência dos direitos
Assistência para direção ecológica	1000ms	500ms	Acesso a internet e disponibilidade do serviço

## 6.1 Cenário Proposto

O ambiente proposto para avaliação consiste em simular movimentações de veículos na cidade de Aracaju-SE Brasil, representando uma área de 174 quilômetros quadrados, transmitindo e recebendo informações dos servidores que compõem a arquitetura I9Vanets.

Foram definidos dois grupos de testes, o primeiro consistiu em avaliar o comportamento da arquitetura referente à quantidade de veículos em conjunto com a limitação das velocidades utilizadas pela telefonia móvel. O segundo, avaliou a capacidade de processamento dos servidores sem limitação da velocidade de acesso.

Cada teste realizado, do primeiro grupo, levou em consideração quantidades diferentes de dispositivos representando os veículos, sendo da seguinte forma: 50, 100, 200 e 400. Para cada faixa, também foi ajustada a largura de banda para definir a velocidade máxima de comunicação de cada dispositivo, seguindo os modelos 2G, 3G, 4G e 5G.

Todo o ambiente, tanto o cliente quanto os servidores, foi montado em máquinas virtuais. No caso dos servidores, foram criados 4 máquinas com linux ubuntu server 16.04 com o postgresql 9.5 e plugin postgis 2.0 e Glassfish 4.1.1 como servidor de aplicação. Todos os servidores possuíam a mesma configuração, sendo 1 GB de Ram e 2 processadores virtuais, e foram executados em uma única máquina física cuja configuração está presente na tabela 19. A disposição dos servidores virtuais foi definida em dois níveis hierárquicos, sendo o servidor nível 0, a raiz da hierarquia, e como filhos, 3 servidores no nível 1.

Para simulação dos clientes, os OBUs e RSUs, foram utilizadas máquinas virtuais com linux ubuntu server 14.04 LTS 32 bits com o Mininet 2.2.1 configurado. Cada rede virtual Mininet instanciou entre 25 e 50 hosts, para que seu desempenho não fosse comprometido. Todas as máquinas com Mininet utilizaram a mesma configuração: 2GB de RAM e 2 processadores virtuais. O modelo do ambiente é mostrado na figura 19.

Foi criado um script em Python, conforme Código 6.1, com intuito de automatizar a criação dos hosts, limitar largura de banda entre 2G, 3G, 4G e 5G, iniciar a execução da aplicação responsável por simular as funcionalidades dos OBUs e RSUs, realizar a comunicação com a arquitetura I9Vanet e registrar em log os dados referentes à comunicação de cada host, em arquivos individuais.

Um host virtual tem a função de representar um veículo real, inclusive com uso de movimentações reais, coletadas por 12 meses, de um sistema de monitoramento de veículos

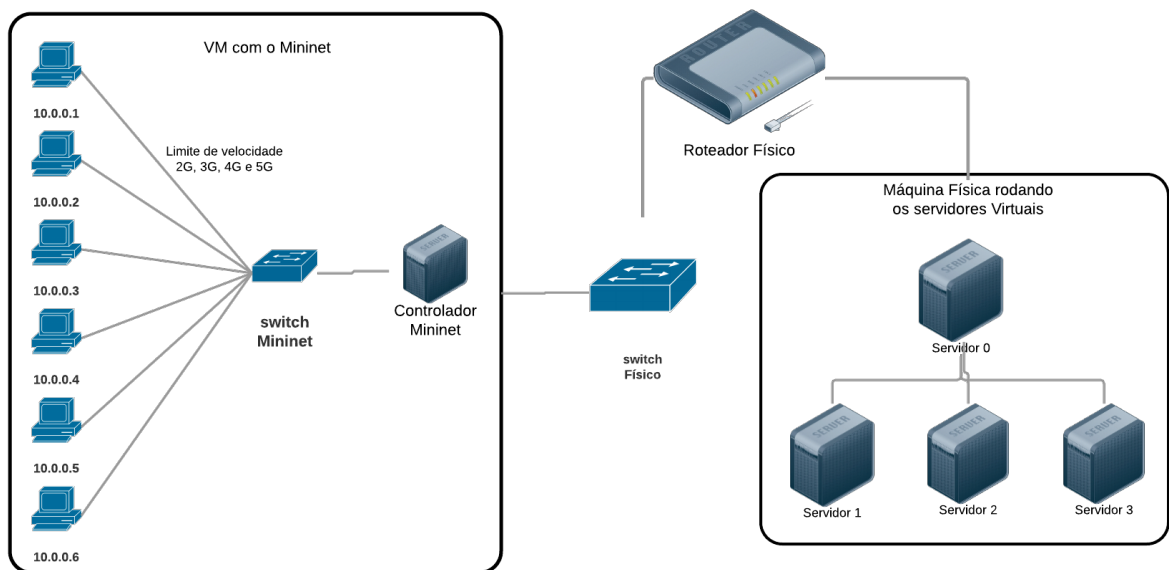


Figura 19 – **Configuração** do ambiente utilizado para a realização dos experimentos. Fonte: Criada pelo autor.

de uma empresa de taxi com 102 carros, totalizando mais de 12 milhões de movimentações. A figura 20 mostra a tela do sistema de monitoramento de uma empresa de taxi.

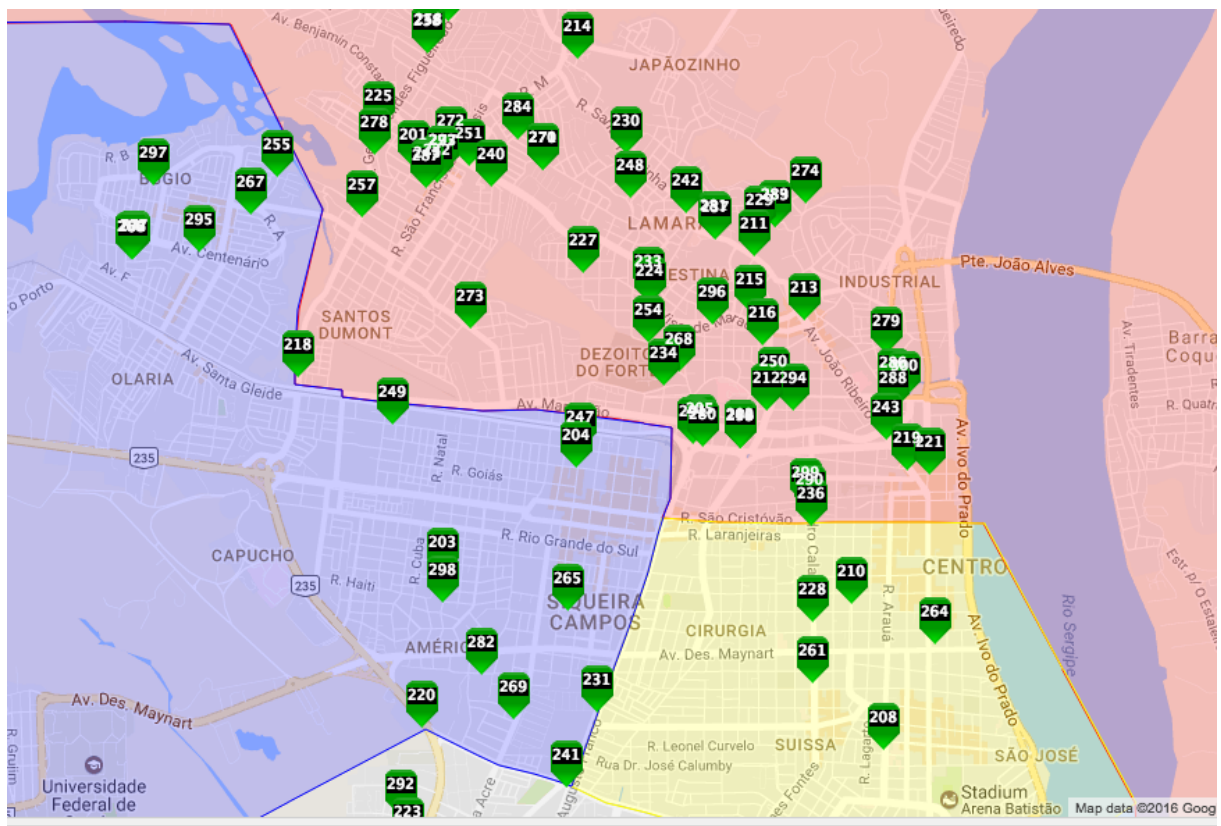


Figura 20 – Tela de monitoramento do Sistema TaxiFast.

A base de movimentação foi extraída separada por veículo e em seguida, armazenada em arquivo e colocada em cada máquina Mininet, para que cada host pudesse simular uma movimentação real. Cada arquivo de movimentação recebeu um número, por exemplo: 1.csv, 2.csv e assim por diante. Então, os host virtuais realizaram um sorteio entre os arquivos disponíveis para efetuar a leitura das informações de movimentação e transmiti-la para o servidor. A tabela 7 mostra exemplo do conteúdo de um arquivo de movimentação.

Latitude	Longitude	Velocidade
-10.9062183	-37.0619264	0
-10.906164143173685	-37.061813870099655	10.7
-10.904381347197752	-37.0671766923126	21.121
-10.904279005332516	-37.067784265675165	26.259
-10.904165425504841	-37.06846398328098	29.678

Tabela 7 – Linhas do arquivo de movimentação.

```

1  #!/usr/bin/python
2  ...
3  class SingleSwitchTopo(Topo):
4  def build(self, n=2):
5      switch = self.addSwitch('s1')
6      #Cria todos os hosts e conecta ao switch
7      for h in range(n):
8          host = self.addHost('h%s' % (h + 1))
9          self.addLink(host, switch)
10 def limit( bw=0.2, cpu=1 ):
11     intf = custom( TCIntf, bw=bw )
12     myTopo = SingleSwitchTopo(n=2)
13     for sched in 'rt', 'cfs':
14         if sched == 'rt':
15             release = quietRun( 'uname -r' ).strip('\r\n')
16             output = quietRun( 'grep CONFIG_RT_GROUP_SCHED /boot/config-%%
17                 s'
18             % release )
19             if output == '# CONFIG_RT_GROUP_SCHED is not set\n':
20                 continue
21             host = custom( CPULimitedHost, sched=sched, cpu=cpu )
22             net = Mininet( topo=myTopo, intf=intf, host=host)
23             net.addNAT().configDefault()
24             net.start()
25             # Para cada host criado, executa a aplicacao de simulacao
26             for host in net.hosts:
27                 if 'h' in host.name:
28                     host.cmd('java -jar SVVMMainTest.jar ' + str(host) + '.
29                         csv /home/mininet/vanet &')
30             CLI( net )
31 if __name__ == '__main__':
32     setLogLevel( 'info' )
33     limit()

```

Listing 6.1 – Script Python para criação dos hosts no mininet.

Para o segundo grupo dos testes, foi utilizado as mesmas máquinas virtuais, com o uso de uma outra aplicação desenvolvida pelo autor. Esta aplicação consistiu em criar inúmeras *threads*, para simular os veículos e realizar a conexão com os servidores sem o controle da largura de banda. Foram executadas em 4 testes, sendo com 800 e 1600 veículos utilizando mensagens abertas e, posteriormente, com informações criptografadas.

## 6.2 Experimentos

Ao iniciar cada experimento, os *hosts* virtuais devem estabelecer a conexão com o servidor nível 0, contudo, se todos os “veículos” iniciarem a conexão ao “mesmo tempo”, ocasionará uma sobrecarga tanto na máquina cliente quanto no servidor. Pensando nisso, foi implementando na aplicação cliente um *delay* forçado antes de iniciar o processo de conexão pela primeira vez. Cada *host* obtém um número pseudo-aleatório entre 0 e 120.000 (2 minutos), o qual será usado para dar uma pausa na aplicação pra depois iniciar o processo de conexão, gerando assim, uma aleatoriedade inicial. Se por algum motivo a conexão do *hosts* cliente cair ou der *timeout*, ela será refeita automaticamente, não considerando mais o tempo de pausa inicial.


As máquinas utilizadas no experimento pertenciam a um laboratório do Instituto Federal de Sergipe - Campus Lagarto e dois notebooks pertencentes ao autor, cuja configurações estão presentes na tabela 8.

Maquina Cliente			
Sist Operacional.	Qnt.	Processador	Memória
Windows 7	12	AMD Phenom II X2 B57 3.20 GHz	4 GB
Mac OS Sierra	1	2,4 GHz Intel Core i5	8 GB
Maquina Servidora			
Sist Operacional.		Processador	Memória
Ubuntu 16.4	1	2,4 GHz Intel Core i7 GHz	8 GB

Tabela 8 – Configuração dos computadores utilizados no experimento.


Para cada cenário, 50,100,200 e 400 veículos, os *hosts* emulados usaram a seguinte distribuição entre as máquinas clientes, como mostra a tabela 9, tendo como preocupação, o poder computacional das máquinas físicas.





Cenário 50		
Máquina Física.	Qnt.	Nr <i>Hosts</i> Cada
Windows	2	25
Cenário 100		
Máquina Física.	Qnt.	Nr <i>Hosts</i> Cada
Windows	4	25
Cenário 200		
Máquina Física.	Qnt.	Nr <i>Hosts</i> Cada
Windows	6	25
Mac OS	1	50
Cenário 400		
Máquina Física.	Qnt.	Nr <i>Hosts</i> Cada
Windows	12	25
Mac OS	2	50

Tabela 9 – Configuração dos computadores utilizados no experimento em cada cenário.



Conforme dito anteriormente, foram definidos 36 cenários de testes, sendo 32 levando em consideração quantidades de veículos, largura de banda da comunicação e a criptografia ou não dos dados e 4 sem o limite de conexão.

### 6.3 Resultados

Inicialmente, os dados colhidos referentes ao RTT, foram avaliados quanto à sua normalidade, para verificar se a distribuição de probabilidade associada às amostras podem ser aproximadas. Para tal, foi utilizado o *Kolmogorov-Smirnov test* (KS) tendo as seguintes hipóteses:

H0: Os dados seguem uma distribuição normal.

Ha: Os dados não seguem uma distribuição normal.

O resultado do teste KS aplicado sobre as médias dos tempos RTT das requisições de cada *host* indica que nenhuma das amostras apresentaram uma distribuição normal a um nível de significância, comumente utilizado, de 0,05. Rejeitando a hipótese H0 para todos os testes com velocidade limitada, como também com 800 veículos sem limite de velocidade.

Já o teste com 1600 veículos, apresentou um *P-Value* acima de 0,05, indicando que a distribuição de probabilidade associada às médias das requisições podem ser aproximadas não rejeitando a hipótese nula, como mostrado na Tabela 10.

Tabela 10 – Resultado do teste de análise de distribuição normal dos dados para cada quantidade de veículos e velocidades de acesso.

Qnt. Veículos	2G	3G	4G	5G	Sem Limite
50	0,0	0,0	0,0	0,0	-
50 encriptado	0,0	0,0	0,0	0,0	-
100	0,0	0,0	0,0	0,0	-
100 encriptado	0,0	0,0	0,0	0,0	-
200	0,0	0,0	0,0	0,0	-
200 encriptado	0,0	0,0	0,0	0,0	-
400	0,0	0,0	0,0	0,0	-
400 encriptado	0,0	0,0	0,0	0,0	-
800	-	-	-	-	0,195
800 encriptado	-	-	-	-	0,310
1600	-	-	-	-	0,801
1600 encriptado	-	-	-	-	0,755

Sob a ótica das variações entre os RTT das requisições, podemos observar que nos cenários com 50 e 100 veículos e utilizando a comunicação aberta, não segura, o coeficiente de variação foi baixo e para os testes acima de 200 veículos houve um crescimento no coeficiente. Isso aconteceu devido à maior quantidade de veículos presentes nos testes e consequentemente, houve mais mudança de domínios. Em média o tempo gasto para realizar a operação *changeServer* é de 2,5 segundos, conforme mostra a tabela 11.

Tabela 11 – Resultado do teste de análise dos coeficientes de variação com mensagens abertas.

Qnt. Veículos	2G	3G	4G	5G
50	0.52360	0.5504	0.5751	0.5435
100	0.5340	0.5490	0.5630	0.5560
200	2.1970	2.5930	2.4360	2.3560
400	1.9220	2.1680	2.4270	2.5360

Todavia, o mesmo experimento utilizando mensagens criptografadas, apresentou coeficientes de variação alto, para todos os cenários, nas velocidades de 2G, 3G e 4G, apenas reduzindo no 5G, com exceção do teste com 400 veículos que apresentou uma diminuição a cada aumento de velocidade, porém continuou alto mesmo com o 5G, como mostra a tabela 12. E vale ressaltar que o tempo medido no envio de mensagens criptografadas,

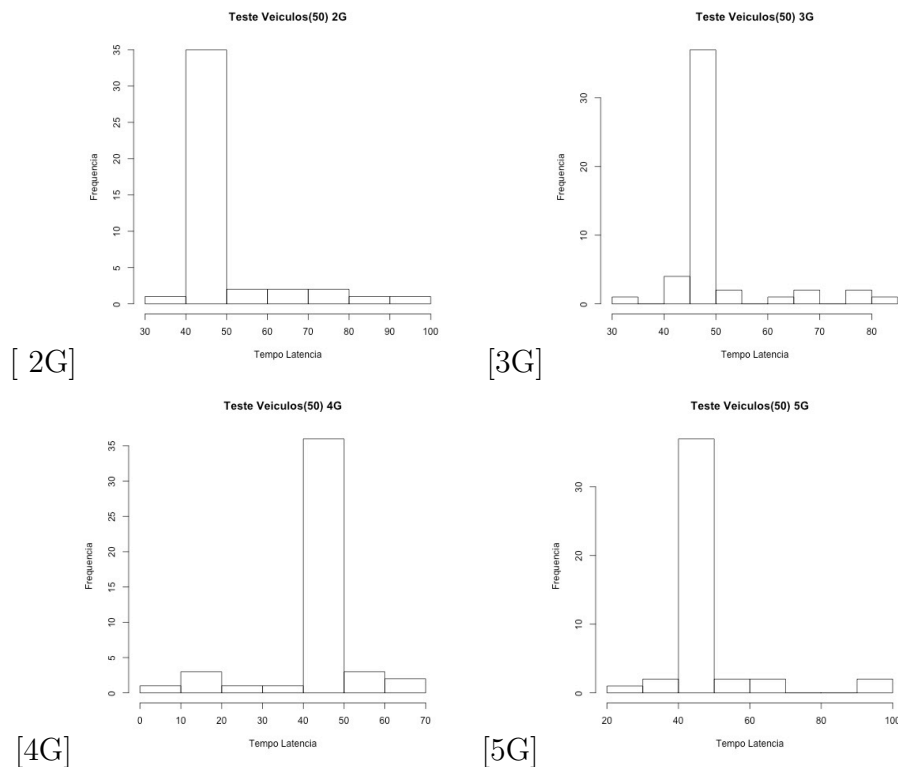
envolve a encriptação na saída do dispositivo, a decrptação na chegada no servidor, a encriptação na resposta do servidor e a decrptação na chegada no dispositivo.

Tabela 12 – Resultado do teste de análise dos coeficientes de variação com mensagens criptografadas.

Qnt. Veículos	2G	3G	4G	5G
50	2.2270	2.0500	1.8631	0.6435
100	2.2520	2.1540	2.5100	0.5560
200	2.0670	1.2260	0.7901	0.6220
400	1.8570	1.6830	1.8230	1.5800

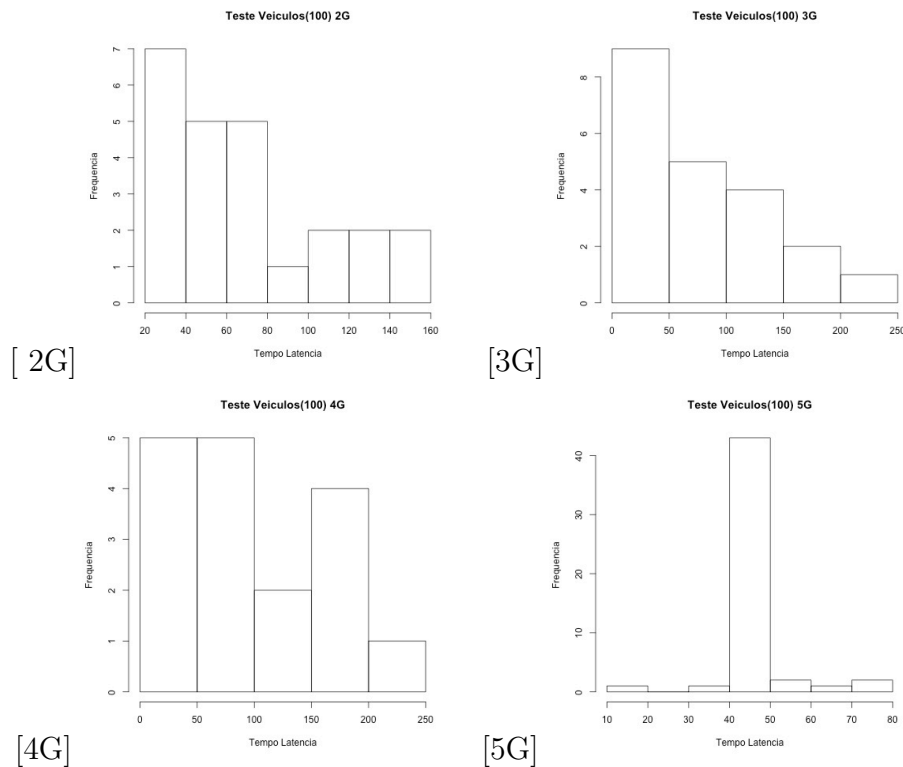
As imagens, contidas na figura 21 mostram o histograma de frequência das médias das requisições para cada velocidade, 2G, 3G, 4G e 5G. Para o teste com 50 veículos com menasgens abertas, observou-se que não há uma relação de melhora nos tempos médios à medida que aumenta a velocidade de 2G para 5G. A concentração dos tempos permanecem na mesma referência, entre 40 e 50 milisegundos.

Figura 21 – Análise de Histograma de Frequência do Teste com 50 veículos com comunicação aberta.



O cenário com 100 veículos apresentou gráficos semelhantes aos de 50, ou seja, o tempo médio das requisições não variou de acordo com o aumento da velocidade, conforme mostra a figura 22.

Figura 22 – Análise de Histograma de Frequência do Teste com 100 veículos com comunicação aberta.



Entretanto, os testes com o cenário de 200 veículos apresentaram uma variação com a diferença de velocidades, de acordo com a figura 23, o teste com a velocidade 2G apresentou maior concentração entre o intervalo de 0 e 50 milissegundos. Já Com a velocidade de 3G, a maior concentração ficou entre 140 e 160 milissegundos.

No experimento com 400, a velocidade 2G apresentou o pior resultado em relação às outras velocidades, contendo, de maneira significativa, médias entre **600 e 700 milissegundos como mostra a figura 24.**

Nas figuras 25, 26 e 27, é feito uma análise sobre os tempos mínimos, médios e máximos das médias dos RTTs das requisições com mensagens abertas e encriptadas, extraídas de cada cenário. Fica evidente que nos cenários de 50, 100 e 200 veículos simultâneos, os tempos mínimos, médios e máximos sofrem pequena variação, entretanto os tempos médios e máximos no cenário com 400 veículos, com velocidade de 2G, há um aumento considerável em relação às outras velocidades, como mostra as figuras 26 e 27, indicando que essa quantidade de veículos já não é indicada com a velocidade e 2G.

No segundo grupo de experimento, foram avaliados as quantidades de 800 e 1600 veículos sem limite da conexão. Conforme figura 28, fica evidente o aumento nos tempos

Figura 23 – Análise de Histograma de Frequência do Teste com 200 veículos com comunicação aberta.

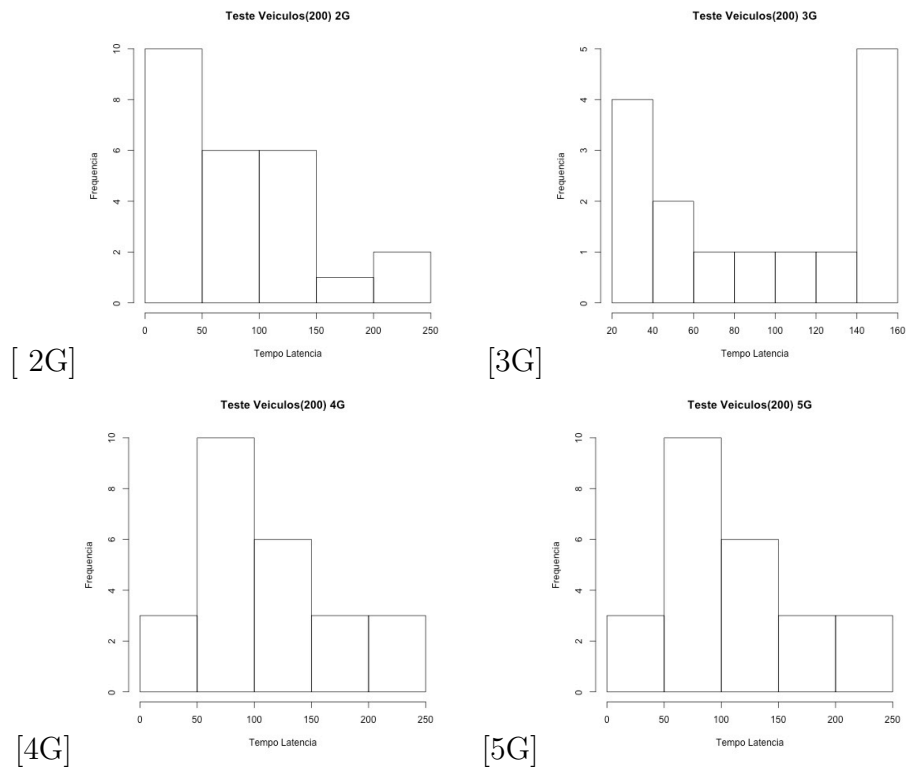
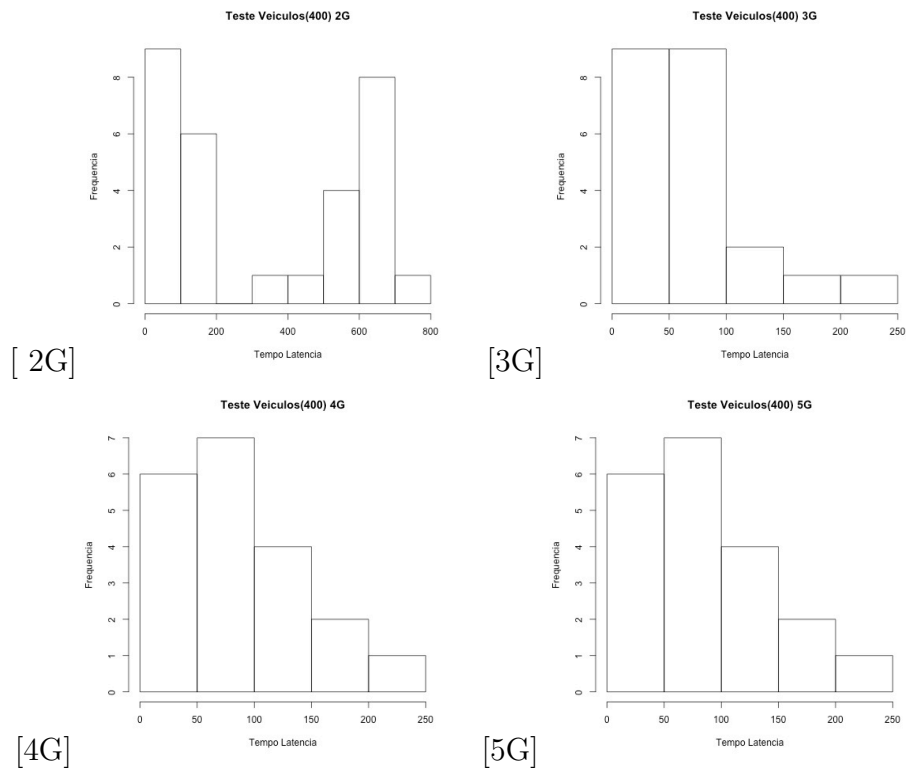


Figura 24 – Análise de Histograma de Frequência do Teste com 400 veículos com comunicação aberta.



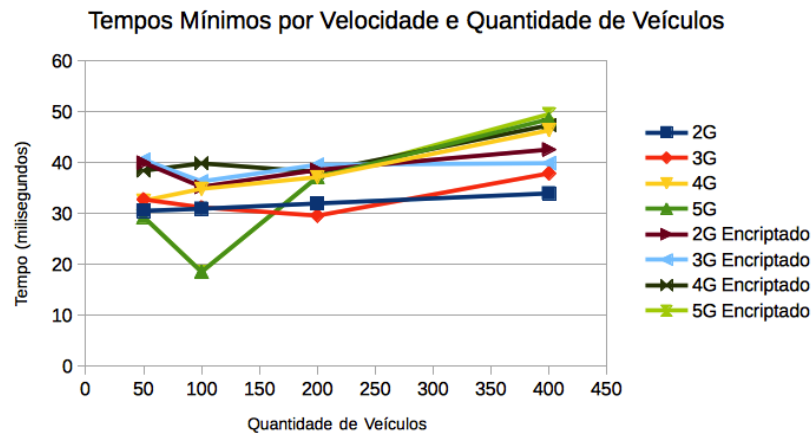


Figura 25 – Gráfico comparativo dos tempos mínimos entres os cenários.

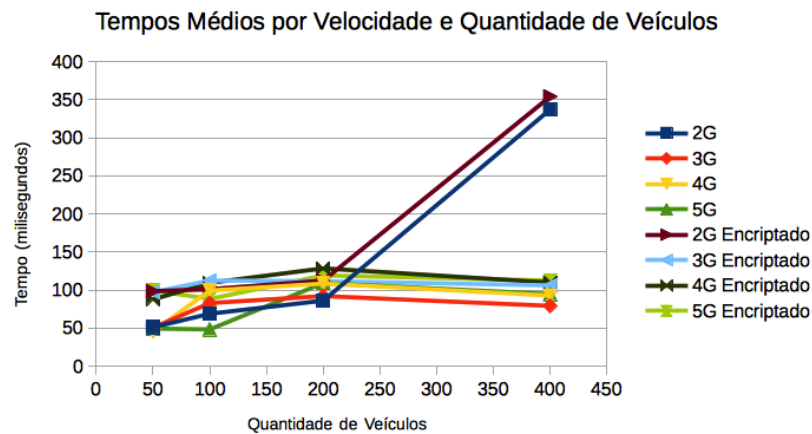


Figura 26 – Gráfico comparativo dos tempos médios entres os cenários.

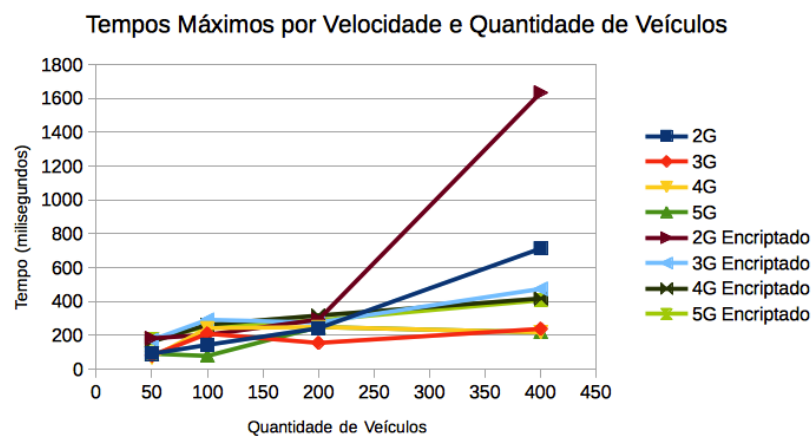
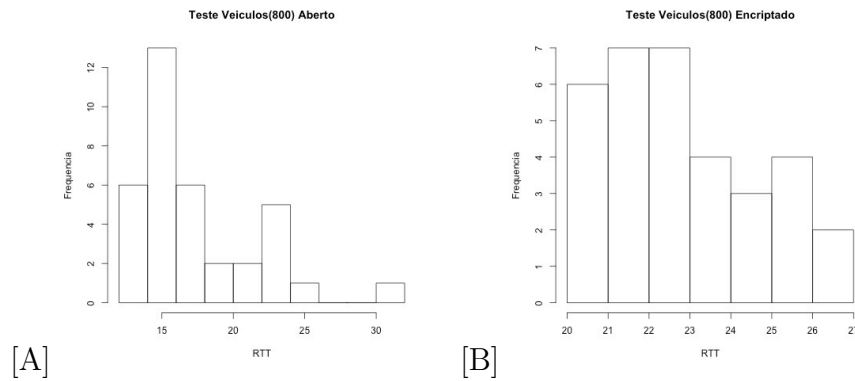


Figura 27 – Gráfico comparativo dos tempos máximos entres os cenários.

médios de RTT das requisições com mensagens criptografadas, figura 28.B, sobre as requisições com conteúdo aberto, figura 28.A. Entretanto, o crescimento não foi significativo

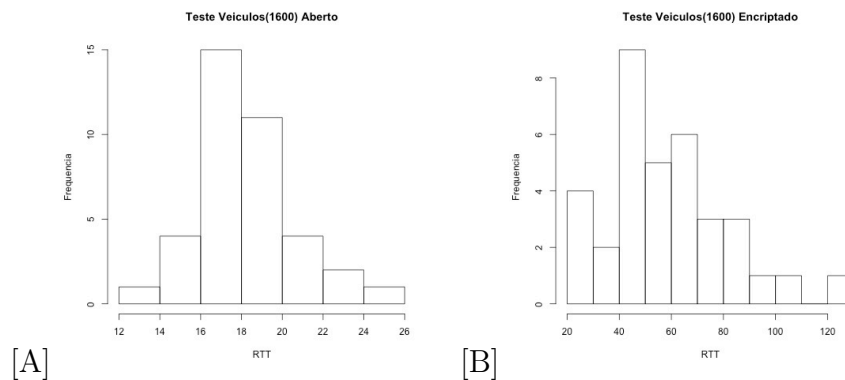
sendo a diferença da maior concentração dos tempos entre 10 e 18 milissegundos, para o teste com mensagens abertas, e 21 e 22 milissegundos, para o teste com conteúdo criptografado, gerando uma diferença média de 8 milissegundos.

Figura 28 – Comparação dos gráficos de histograma para os RTTs com 800 veículos.



Para o teste com 1600 veículos, houve um aumento considerável dos tempos médios das requisições como mostra a figura 29. A concentração dos tempos médios na figura 29.A, ficou entre 16 e 20 milissegundos, já no teste com mensagens criptografadas, ficou em 40 e 90 milissegundos conforme figura 29.B.

Figura 29 – Comparação dos gráficos de histograma para os RTTs com 1600 veículos.



Na figura 30, é feito uma análise sobre os tempos mínimos, médios e máximos das médias dos RTTs das requisições com mensagens abertas e encriptadas, extraídas dos cenários com 800 e 1600 veículos. De acordo com o gráfico, o tempo médio e máximo para o teste com 1600 dispositivos mostrou um aumento ao utilizar mensagens criptografadas indicando um salto e o limite da arquitetura.



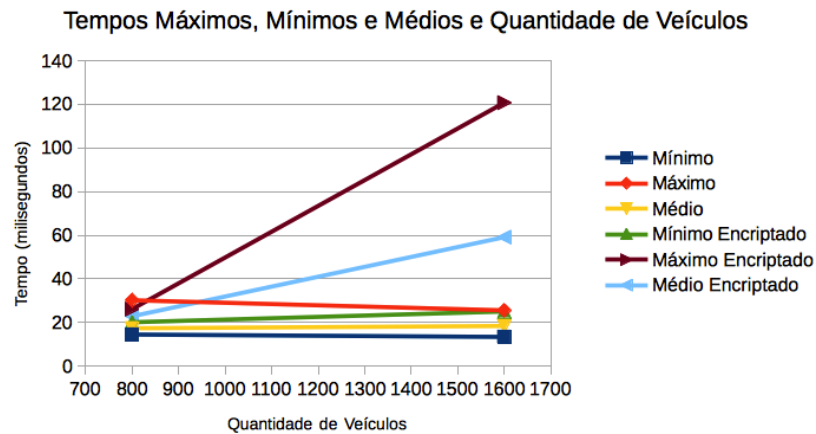


Figura 30 – Gráfico comparativo dos tempos mínimos entre os cenários 800 e 1600.

### 6.3.1 Taxa de Transferência

Os veículos enviam, a cada 10 segundos, os dados com localização, direção e velocidade, estas informações variaram entre 212 e 252 bytes e a tabela 13 mostra as velocidades de transmissão utilizadas por cada veículo nos experimentos e o gráfico da figura 31 exibe a capacidade estimada ocupada por cada teste executado com limites de conexões.

Tabela 13 – Velocidade de cada link por veículo definidos de acordo com Li et al. (2009).

Link	Largura de Banda(kbps)
2G	400
3G	2.000
4G	100.000
5G	10.000.000

### 6.3.2 Processamento

Foi avaliado o nível de processamento dos servidores, sendo que o servidor nível 0 chegou a 100% de processamento nos primeiros 2 minutos, no teste com 800 veículos, e depois reduziu para 20%, onde se manteve, conforme figura 32. Isso ocorreu devido a todas as conexões iniciais serem feitas para este servidor, como também quando há necessidade de mudança de domínio, o servidor nível 0 é requisitado de acordo com a hierarquia estabelecida.



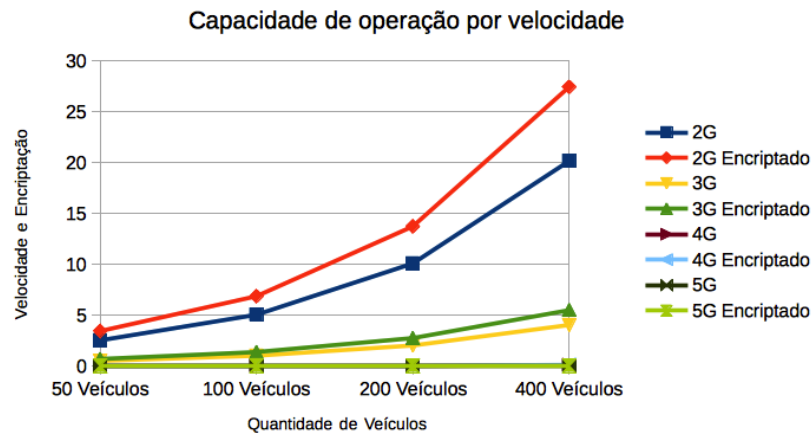


Figura 31 – Comparativo da capacidade de cada *link* pela quantidade de veículos.

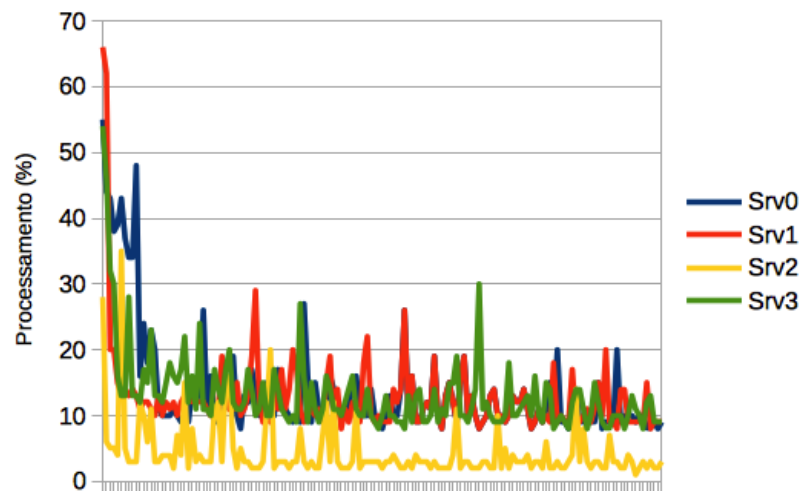


Figura 32 – Nível de processamento dos servidores para o teste com 800 veículos com conteúdo criptografado.

No teste com 1600 veículos, utilizando conteúdo criptografado, o processamento chegou a 100% no servidor nível 0 e **permaneceu alto** durante todo o teste, como mostra a figura 33.

### 6.3.3 Perda de Pacotes

Durante os experimentos foram analisados as perdas em cada dispositivo por meio de dois contadores. O primeiro foi utilizado para contabilizar as requisições realizadas e outro para apurar a resposta das requisições. Como mostrado na 34.A, a maior perda ocorre nas velocidades de 2G utilizando encriptação atingindo o valor a cima de 25% de perda. Contudo, para o teste com 400 veículos, a perda ficou abaixo de 15%. Já as perdas

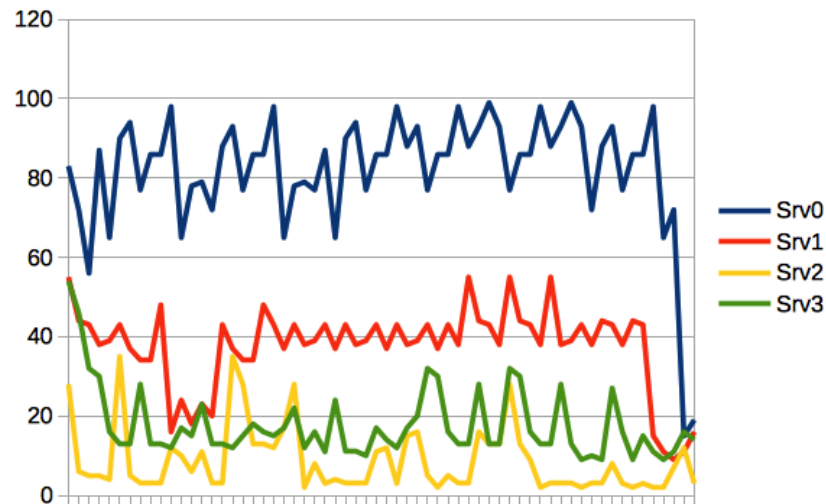
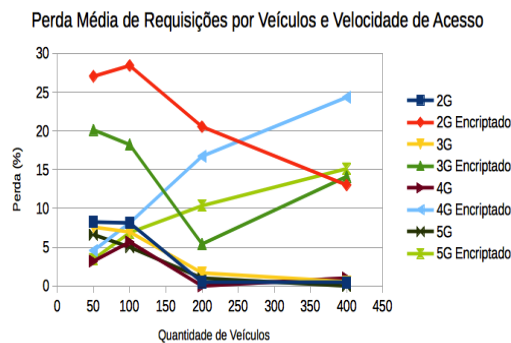


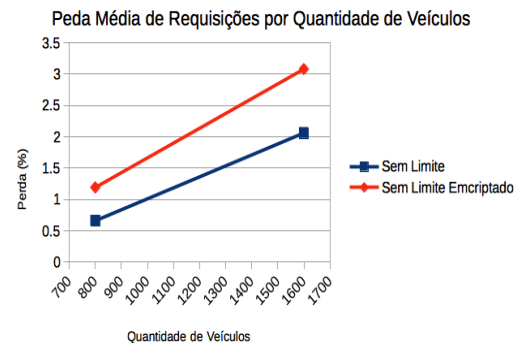
Figura 33 – Nível de processamento dos servidores para o teste com 1600 veículos com conteúdo criptografado.

para os experimentos sem limite de conexão, apresentaram valor abaixo de 1% para o teste com 800 veículos e 2.06% para o teste com 1600 veículos e utilizando mensagens criptografadas, ambos ficaram mais altos em relação à comunicação aberta, como mostra a figura 34.B.

Figura 34 – Percentual de perda para cada experimento realizado..



[A]



[B]

## 7 Conclusão e Trabalhos Futuros

### 7.1 Conclusões deste Trabalho

O presente trabalho apresentou o modelo de uma arquitetura de *software* para rede veiculares em nuvem, denominado I9Vanets, como também, uma avaliação com intuito de medir a sua eficácia e eficiência, visando atender às demandas e desafios relacionados à VANETs. Este modelo permite montar uma **rede veicular em nuvem** e realizar todo gerenciamento e comunicação de maneira virtual, tornando os equipamentos OBUs e RSUs mais simples e baratos.

Foram realizados alguns testes estatísticos, com objetivo de analisar o comportamento da arquitetura. Sobre o teste da normalidade da distribuição, foi constatado que **todos os testes com limite de conexão não apresentou uma distribuição normal**, já o teste **sem o controle de conexão de 1600 veículos, mostrou que a distribuição de probabilidade associada às médias das requisições podem ser aproximadas**. Outra análise realizada, foi relacionado à dispersão dos dados através do coeficiente de variação, o qual avaliou, sobre as amostras de RTT, que a dispersão em cada teste, define um comportamento semelhante, porém o coeficiente de variação medido nos testes com controle do *link*, apresentou **grandes variações** para a troca de mensagens criptografadas, sugerindo que o processo de encriptação e deciptação está afetando as requisições.

Em relação ao nível de processamento dos servidores, o servidor nível 0 é o responsável pelas primeiras conexões feitas pelos veículos e durante a mudança de domínios. Pensando nisso, para uma melhor organização e diminuição de carga em um único servidor, é interessante que os domínios vizinhos fiquem, ao máximo, no mesmo nível de hierarquia tendo o mesmo servidor pai. Assim, essa sub-árvore responderia de maneira mais eficiente ao comando *changeServer* e diminuiria a pressão sobre o servidor raiz.

Uma outra possibilidade é que o servidor nível 0 seja utilizado apenas para coordenar a estrutura, retirando de si o papel de gerenciar os veículos. Assim ele teria o papel de receber as primeiras conexões e encaminhá-las para o servidor correspondente, o mesmo na operação *changeServer*.

Os testes aplicados à arquitetura, com 800 e 1600 veículos, indicaram que o limite da organização proposta está em 1600 veículos e que mais veículos conectados a esta plataforma poderá comprometer o tempo de resposta.

## 7.2 Trabalhos Futuros

Os resultados e contribuições apresentados nesta dissertação criam oportunidades para trabalhos futuros: por meio de mudança na organização dos servidores visando uma melhor distribuição dos veículos e diminuindo a carga com a operação *ChangeServer*; criação de novos algoritmos de roteamentos; novos protocolos de comunicação; novas regras de segurança; implementação de uma plataforma web de simulação, facilitando a utilização em ambiente reais; criando diversas aplicações como sistema de detecção e alerta de congestionamento de trânsito, permitindo um RSU avisar a outros RSU e/ou OBU; Controle de passagem livre para veículos de urgência e emergência.

## Referências

- AL-KAHTANI, M. S. Survey on security attacks in vehicular ad hoc networks (vanets). In: IEEE. *Signal Processing and Communication Systems (ICSPCS), 2012 6th International Conference on*. [S.l.], 2012. p. 1–9. Citado 2 vezes nas páginas 40 e 41.
- AVELAR, E. et al. Interoperability issues on heterogeneous wireless communication for smart cities. *Computer Communications*, Elsevier, v. 58, p. 4–15, 2015. Citado na página 39.
- BALL, R.; DULAY, N. Enhancing traffic intersection control with intelligent objects. *Urban Internet of Things Towards Programmable Realtime Cities*, 2010. Citado na página 13.
- BARBA, C. T.; AGUIRRE, K. O.; IGARTUA, M. A. Performance evaluation of a hybrid sensor and vehicular network to improve road safety. In: ACM. *Proceedings of the 7th ACM workshop on Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks*. [S.l.], 2010. p. 71–78. Citado na página 24.
- BIRK, W.; OSIPOV, E.; ELIASSON, J. iroad—cooperative road infrastructure systems for driver support. In: *Proceedings of the 16th ITS World Congress*. [S.l.: s.n.], 2009. Citado na página 25.
- BLUM, J.; ESKANDARIAN, A.; HOFFMAN, L. Mobility management in ivc networks. In: IEEE. *Intelligent Vehicles Symposium, 2003. Proceedings. IEEE*. [S.l.], 2003. p. 150–155. Citado na página 46.
- BOUKERCHE, A. Performance evaluation of routing protocols for ad hoc wireless networks. *Mobile Networks and Applications*, Springer-Verlag New York, Inc., v. 9, n. 4, p. 333–342, 2004. Citado na página 44.
- BRIESEMEISTER, L.; SCHAFERS, L.; HOMMEL, G. Disseminating messages among highly mobile hosts based on inter-vehicle communication. In: IEEE. *Intelligent Vehicles Symposium, 2000. IV 2000. Proceedings of the IEEE*. [S.l.], 2000. p. 522–527. Citado na página 47.
- BUBENIKOVA, E.; DURECH, J.; FRANEKOVA, M. Security solutions of intelligent transportation system's applications with using vanet networks. In: IEEE. *Control Conference (ICCC), 2014 15th International Carpathian*. [S.l.], 2014. p. 63–68. Citado na página 36.
- CARVALHO, C. H. R. d. et al. Mobilidade urbana e posse de veículos: análise da pnad 2009. Instituto de Pesquisa Econômica Aplicada (Ipea), 2010. Citado na página 13.
- CAVALCANTI, S. R. *Veer: Um algoritmo de seleção de pares em redes ad hoc veiculares*. Tese (Doutorado) — UNIVERSIDADE FEDERAL DO RIO DE JANEIRO, 2008. Citado 2 vezes nas páginas 14 e 15.
- CHEN, W. et al. Dynamic local peer group organizations for vehicle communications. In: IEEE. *Mobile and Ubiquitous Systems-Workshops, 2006. 3rd Annual International Conference on*. [S.l.], 2006. p. 1–8. Citado na página 15.

COULOURIS, G. et al. *Sistemas Distribuídos-: Conceitos e Projeto*. [S.l.]: Bookman Editora, 2013. Citado 3 vezes nas páginas 28, 29 e 31.

CRISTIAN, F. Reaching agreement on processor-group membership in synchronous distributed systems. *Distributed Computing*, Springer, v. 4, n. 4, p. 175–187, 1991. Citado na página 31.

DANTAS, A. *TRÂNSITO: O JOGO DO CAOS*. 2011. Disponível em: <http://www.autoentusiastasclassic.com.br/2011/05/transito-o-jogo-do-caos.html>. Citado 3 vezes nas páginas 5, 14 e 49.

DAS, B.; BHARGHAVAN, V. Routing in ad-hoc networks using minimum connected dominating sets. In: IEEE. *Communications, 1997. ICC'97 Montreal, Towards the Knowledge Millennium. 1997 IEEE International Conference on*. [S.l.], 1997. v. 1, p. 376–380. Citado na página 46.

DIB, R. P. E. et al. A systematic review of the interventions to promote the wearing of hearing protection. *São Paulo Medical Journal*, SciELO Brasil, v. 125, n. 6, p. 359–361, 2007. Citado na página 13.

DURRESI, M.; DURRESI, A.; BAROLLI, L. Emergency broadcast protocol for inter-vehicle communications. In: IEEE. *11th International Conference on Parallel and Distributed Systems (ICPADS'05)*. [S.l.], 2005. v. 2, p. 402–406. Citado na página 47.

ELTOWEISSY, M.; OLARIU, S.; YOUNIS, M. Towards autonomous vehicular clouds. In: SPRINGER. *International Conference on Ad Hoc Networks*. [S.l.], 2010. p. 1–16. Citado na página 17.

ENGOULOU, R. G. et al. Vanet security surveys. *Computer Communications*, Elsevier, v. 44, p. 1–13, 2014. Citado na página 38.

FALCHETTI, A.; AZURDIA-MEZA, C.; CESPEDES, S. Vehicular cloud computing in the dawn of 5g. In: IEEE. *2015 CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON)*. [S.l.], 2015. p. 301–305. Citado 3 vezes nas páginas 16, 17 e 20.

FISCHER, M. J.; LYNCH, N. A.; PATERSON, M. S. Impossibility of distributed consensus with one faulty process. *Journal of the ACM (JACM)*, ACM, v. 32, n. 2, p. 374–382, 1985. Citado na página 31.

FÜSSLER, H. et al. Mobicom poster: location-based routing for vehicular ad-hoc networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, ACM, v. 7, n. 1, p. 47–49, 2003. Citado na página 45.

GORENDER, S.; MACÊDO, R. Um modelo para tolerância a falhas em sistemas distribuídos com qos. *Anais do Simpósio Brasileiro de Redes de Computadores, SBRC 2002*, p. 277–292, 2002. Citado 2 vezes nas páginas 30 e 31.

GRAY, N. A. Comparison of web services, java-rmi, and corba service implementations. In: *Proceedings of the 5th Australasian Workshop on Software and System Architectures at ASWEC*. [S.l.: s.n.], 2004. p. 52–63. Citado 3 vezes nas páginas 5, 8 e 33.

GUPTE, S.; YOUNIS, M. Vehicular networking for intelligent and autonomous traffic management. In: IEEE. *Communications (ICC), 2012 IEEE International Conference on*. [S.l.], 2012. p. 5306–5310. Citado na página 14.

HADALLER, D. et al. Vehicular opportunistic communication under the microscope. In: ACM. *Proceedings of the 5th international conference on Mobile systems, applications and services*. [S.l.], 2007. p. 206–219. Citado na página 15.

HAJJI, T.; BARGAOUI, H. Design of a vanet testbed based on cloud computing. Citado na página 17.

HALL, G. B.; LEAHY, M. G. *Open source approaches in spatial data handling*. [S.l.]: Springer, 2008. v. 2. Citado na página 61.

HUSSAIN, R. et al. Rethinking vehicular communications: Merging vanet with cloud computing. In: IEEE. *Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on*. [S.l.], 2012. p. 606–609. Citado na página 17.

ICE, R. et al. *REGIONAL ITS ARCHITECTURE GUIDANCE: DEVELOPING, USING, AND MAINTAINING AN ITS ARCHITECTURE FOR YOUR REGION*. [S.l.], 2001. Citado na página 13.

ISAAC, J. T.; ZEADALLY, S.; CAMARA, J. S. Security attacks and solutions for vehicular ad hoc networks. *IET communications*, IET, v. 4, n. 7, p. 894–903, 2010. Citado 2 vezes nas páginas 41 e 42.

JACQUET, P. et al. Optimized link state routing protocol for ad hoc networks. In: IEEE. *Multi Topic Conference, 2001. IEEE INMIC 2001. Technology for the 21st Century. Proceedings. IEEE International*. [S.l.], 2001. p. 62–68. Citado na página 44.

JAKUBIAK, J.; KOUCHERYAVY, Y. State of the art and research challenges for vanets. In: IEEE. *Consumer communications and networking conference, 2008. CCNC 2008. 5th IEEE*. [S.l.], 2008. p. 912–916. Citado na página 36.

KARP, B.; KUNG, H.-T. Gpsr: Greedy perimeter stateless routing for wireless networks. In: ACM. *Proceedings of the 6th annual international conference on Mobile computing and networking*. [S.l.], 2000. p. 243–254. Citado na página 45.

KIHL, M.; SICHITIU, M.; JOSHI, H. Design and evaluation of two geocast protocols for vehicular ad-hoc networks. *Journal of Internet Engineering*, Klidarithmos Press, 2008. Citado na página 47.

LAMPORT, L.; SHOSTAK, R.; PEASE, M. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, ACM, v. 4, n. 3, p. 382–401, 1982. Citado na página 31.

LEE, E. et al. Vehicular cloud networking: architecture and design principles. *IEEE Communications Magazine*, IEEE, v. 52, n. 2, p. 148–155, 2014. Citado 3 vezes nas páginas 5, 17 e 18.

LEONTIADIS, I.; MASCOLO, C. Geopps: Geographical opportunistic routing for vehicular networks. In: IEEE. *2007 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*. [S.l.], 2007. p. 1–6. Citado na página 45.



- LI, C. Travel information service system for public travel based on soa. In: IEEE. *Service Operations and Logistics and Informatics (SOLI), 2010 IEEE International Conference on*. [S.l.], 2010. p. 321–324. Citado na página [13](#).
- LI, F.; WANG, Y. Routing in vehicular ad hoc networks: A survey. *IEEE Vehicular technology magazine*, IEEE, v. 2, n. 2, p. 12–22, 2007. Citado na página [45](#).
- LI, L.; LIU, Y.-A.; TANG, B.-H. Snms: an intelligent transportation system network architecture based on wsn and p2p network. *The journal of China universities of posts and telecommunications*, Elsevier, v. 14, n. 1, p. 65–70, 2007. Citado na página [25](#).
- LI, X. et al. The future of mobile wireless communication networks. In: IEEE. *Communication Software and Networks, 2009. ICCSN'09. International Conference on*. [S.l.], 2009. p. 554–557. Citado 2 vezes nas páginas [8](#) e [79](#).
- LIANG, W. et al. Vehicular ad hoc networks: architectures, research issues, methodologies, challenges, and trends. *International Journal of Distributed Sensor Networks*, Taylor & Francis, Inc., v. 2015, p. 17, 2015. Citado na página [15](#).
- LIU, Y.-C.; CHEN, C.; CHAKRABORTY, S. A software defined network architecture for geobroadcast in vanets. In: IEEE. *2015 IEEE International Conference on Communications (ICC)*. [S.l.], 2015. p. 6559–6564. Citado na página [16](#).
- LOSILLA, F. et al. A comprehensive approach to wsn-based its applications: A survey. *Sensors*, Molecular Diversity Preservation International, v. 11, n. 11, p. 10220–10265, 2011. Citado 7 vezes nas páginas [5](#), [13](#), [22](#), [23](#), [24](#), [25](#) e [26](#).
- LUÍS, N. M. A. *Melhoria de protocolos de encaminhamento em VANETs de alta densidade*. Tese (Doutorado) — FCT-UNL, 2009. Citado 10 vezes nas páginas [5](#), [8](#), [13](#), [36](#), [37](#), [43](#), [44](#), [46](#), [48](#) e [49](#).
- LYNCH, N. A. *Distributed algorithms*. [S.l.]: Morgan Kaufmann, 1996. Citado na página [31](#).
- MACÊDO, R. J. de A. Failure detection in asynchronous distributed systems. 2000. Citado na página [31](#).
- MATOS, L. B. C. d. et al. Análise de desempenho de algoritmos criptográficos assimétricos em uma rede veicular (vanet). *Ciência da Computação*, 2013. Citado 2 vezes nas páginas [16](#) e [38](#).
- MEJRI, M. N.; BEN-OTHTMAN, J.; HAMDI, M. Survey on vanet security challenges and possible cryptographic solutions. *Vehicular Communications*, Elsevier, v. 1, n. 2, p. 53–66, 2014. Citado 4 vezes nas páginas [40](#), [41](#), [42](#) e [43](#).
- MELNIKOV, A.; FETTE, I. The websocket protocol. *Request for Comments*, v. 6455, 2011. Citado na página [34](#).
- MERSHAD, K.; ARTAIL, H. Finding a star in a vehicular cloud. *IEEE Intelligent transportation systems magazine*, IEEE, v. 5, n. 2, p. 55–68, 2013. Citado na página [17](#).
- MIURA, S.; ZHAN, Y.; KURODA, T. Evaluation of parking search using sensor network. In: IEEE. *2006 1st International Symposium on Wireless Pervasive Computing*. [S.l.], 2006. p. 6–pp. Citado na página [25](#).



NANDAN, A. et al. Co-operative downloading in vehicular ad-hoc wireless networks. In: IEEE. *Second Annual Conference on Wireless On-demand Network Systems and Services*. [S.l.], 2005. p. 32–41. Citado na página 15.

NASIM, R.; KASSLER, A. Distributed architectures for intelligent transport systems: A survey. In: IEEE. *Network Cloud Computing and Applications (NCCA), 2012 Second Symposium on*. [S.l.], 2012. p. 130–136. Citado na página 13.

NIEDERMAIER, B. et al. The new bmw idrive—applied processes and methods to assure high usability. In: SPRINGER. *International Conference on Digital Human Modeling*. [S.l.], 2009. p. 443–452. Citado na página 26.

PAPADIMITRATOS, P. et al. Secure vehicular communication systems: design and architecture. *IEEE Communications Magazine*, IEEE, v. 46, n. 11, p. 100–109, 2008. Citado 2 vezes nas páginas 8 e 66.

PATHRE, A. Identification of malicious vehicle in vanet environment from ddos attack. *Journal of Global Research in Computer Science*, v. 4, n. 6, p. 30–34, 2013. Citado na página 42.

PEDEN, M. et al. *World report on road traffic injury prevention*. [S.l.]: World Health Organization Geneva, 2004. Citado na página 13.

QIAN, Y.; LU, K.; MOAYERI, N. A secure vanet mac protocol for dsrc applications. In: IEEE. *IEEE GLOBECOM 2008-2008 IEEE Global Telecommunications Conference*. [S.l.], 2008. p. 1–5. Citado 2 vezes nas páginas 5 e 50.

QIN, H. et al. An integrated network of roadside sensors and vehicles for driving safety: Concept, design and experiments. In: IEEE. *Pervasive Computing and Communications (PerCom), 2010 IEEE International Conference on*. [S.l.], 2010. p. 79–87. Citado na página 25.

QIN, Y.; HUANG, D.; ZHANG, X. Vehicloud: Cloud computing facilitating routing in vehicular networks. In: IEEE. *2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications*. [S.l.], 2012. p. 1438–1445. Citado na página 17.

RAWAT, A.; SHARMA, S.; SUSHIL, R. Vanet: Security attacks and its possible solutions. *Journal of Information and Operations Management*, Bioinfo Publications, v. 3, n. 1, p. 301, 2012. Citado na página 41.

RAYA, M.; PAPADIMITRATOS, P.; HUBAUX, J.-P. Securing vehicular communications. *IEEE Wireless Communications Magazine, Special Issue on Inter-Vehicular Communications*, v. 13, n. LCA-ARTICLE-2006-015, p. 8–15, 2006. Citado 2 vezes nas páginas 38 e 43.

SAMARA, G.; AL-SALIH, W. A.; SURES, R. Security issues and challenges of vehicular ad hoc networks (vanet). In: IEEE. *New Trends in Information Science and Service Science (NISS), 2010 4th International Conference on*. [S.l.], 2010. p. 393–398. Citado 2 vezes nas páginas 16 e 38.

SANTOS, R.; EDWARDS, R.; EDWARDS, A. Cluster-based location routing algorithm for inter-vehicle communication. In: IEEE. *Vehicular Technology Conference, 2004. VTC2004-Fall. 2004 IEEE 60th*. [S.l.], 2004. v. 2, p. 914–918. Citado na página 46.

SOLINGEN, R. V. et al. Goal question metric (gqm) approach. *Encyclopedia of software engineering*, Wiley Online Library, 2002. Citado na página 65.

SOUZA RONIEL DE; SOARES, A. C. B. Estimativa e sinalização de congestionamentos de tráfego através de redes veiculares v2v. *Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, 2014. Citado na página 37.

SUMRA, I. A.; HASBULLAH, B.; MANAN, J. Comparative study of security hardware modules (edr, tpd and tpm) in vanet. In: *Third National Information Technology Symposium*, NITS 2011 Symposium. [S.l.: s.n.], 2011. p. 6–9. Citado na página 13.

SUNG, K.; YOO, J.; KIM, D. Collision warning system on a curved road using wireless sensor networks. In: IEEE. *2007 IEEE 66th Vehicular Technology Conference*. [S.l.], 2007. p. 1942–1946. Citado na página 24.

SYSTEMATICS, C.; MEYER, M. Crashes vs. congestion-what’s the cost to society. *Prepared for the American Automobile Association*, 2011. Citado na página 13.

TANENBAUM, A.; STEEN, M. V. *Sistemas Distribuídos - Princípios e Paradigmas*. 2ª Edição. [S.l.]: Editora Pearson Prentice Hall, 2007. Citado na página 28.

TANGADE, S. S.; MANVI, S. S. A survey on attacks, security and trust management solutions in vanets. In: IEEE. *Computing, Communications and Networking Technologies (ICCCNT), 2013 Fourth International Conference on*. [S.l.], 2013. p. 1–6. Citado 3 vezes nas páginas 39, 40 e 42.

THEMUDO, V. M. Implementação de um servidor de negociação em bolsa baseado em websocket. 2014. Citado 3 vezes nas páginas 5, 34 e 35.

TTI, T. T. I. U. M. *Texas Transport Institute Urban Mobility Report 2014*. 2014. Disponível em: <http://d2dtl5nnlpr0r.cloudfront.net/tti.tamu.edu/documents/ums/congestion-data/national/national-table1.pdf>. Citado na página 13.

WANGHAM, M. S. et al. Segurança em redes veiculares: Inovações e direções futuras. 2014. Citado 7 vezes nas páginas 5, 38, 39, 40, 41, 42 e 43.

WASEF, A. et al. Complementing public key infrastructure to secure vehicular ad hoc networks [security and privacy in emerging wireless networks]. *IEEE Wireless Communications*, IEEE, v. 17, n. 5, p. 22–28, 2010. Citado na página 43.

WEINGÄRTNER, E.; KARGL, F. A prototype study on hybrid sensor-vehicular networks. *Proceedings of the 6th GI/ITG KuVS Fachgespräch “Wireless Sensor Networks*, 2007. Citado na página 24.

XU, Q. et al. Design and analysis of highway safety communication protocol in 5.9 ghz dedicated short range communication spectrum. In: IEEE. *Vehicular Technology Conference, 2003. VTC 2003-Spring. The 57th IEEE Semiannual*. [S.l.], 2003. v. 4, p. 2451–2455. Citado na página 13.

YAN, G. et al. Security challenges in vehicular cloud computing. *IEEE Transactions on Intelligent Transportation Systems*, IEEE, v. 14, n. 1, p. 284–294, 2013. Citado na página 17.

ZHANG, J. A survey on trust management for vanets. In: IEEE. *2011 IEEE International Conference on Advanced Information Networking and Applications*. [S.l.], 2011. p. 105–112. Citado na página [43](#).

ZHU, T.; LIU, Z. Intelligent transport systems in china: Past, present and future. In: IEEE. *Measuring Technology and Mechatronics Automation (ICMTMA), 2015 Seventh International Conference on*. [S.l.], 2015. p. 581–584. Citado na página [13](#).