GIULIANA BARATTO BARONE

Mecanismos para emissão de certificados com privacidade em redes veiculares

São Paulo
2018

GIULIANA BARATTO BARONE

Mecanismos para emissão de certificados com privacidade em redes veiculares

Trabalho apresentado à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Bacharel em Engenharia.

São Paulo
2018

GIULIANA BARATTO BARONE

Mecanismos para emissão de certificados com privacidade em redes veiculares

Trabalho apresentado à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Bacharel em Engenharia.

Área de Concentração:

Computação e Sistemas Digitais

Orientador:

Prof. Dr. Marcos Antonio Simplicio Junior

São Paulo
2018

Nome: BARONE, Giuliana Baratto

Título: Mecanismos para emissão de certificados com privacidade em redes veiculares

Monografia (Trabalho de Conclusão de Curso) apresentada à Escola Politécnica da Universidade de São Paulo para a obtenção do Título de Bacharel em Engenharia na área de Engenharia de Computação.

Trabalho entregue para o departamento em:

Responsáveis

Prof. Dr. Marcos Antonio Simplicio Junior
(Orientador)

Giuliana Baratto Barone
(Orientada)

AGRADECIMENTOS

Agradeço aos meus amigos e à minha família por terem me apoiado durante estes últimos cinco anos. Agradecimentos especiais aos pais, que me auxiliaram na escolha do curso, à minha irmã Carolina, que manteve minha sanidade, e à minha amiga Larissa, por não ter desistido de mim e ter me emprestado seu computador para a realização dos testes. Agradeço também a Leonardo Hirano, por aguentar meus momentos de desespero na produção deste trabalho.

Agradeço a todos os professores que me acompanharam em minha jornada politécnica e a tornaram um pouco mais agradável. Agradeço também a meu orientador por me guiar ao longo deste ano na conclusão do projeto.

"You should always waste time when you don't have any. Time is not the boss of you."

-Doctor Who (2005), 6ª Temporada, Episódio 14

RESUMO

O advento de veículos com características autônomas e semi-autônomas é acompanhado pela promessa de melhorias quanto a prevenção de acidentes, redução de tempo de viagem, entre outros benefícios. No entanto, apenas a automação do veículo não é capaz de proporcionar os avanços mencionados, sendo necessária a troca de informações assinadas entre os usuários. Este documento aborda os principais requisitos para a emissão de certificados em redes veiculares, com discussão sobre os estudos mais importantes do meio. Dentre artigos recentes, é dado destaque à solução *Security Credential Management System* (SCMS), que representa hoje a principal candidata à padronização nos Estados Unidos. O objetivo deste trabalho é analisar a adequabilidade do sistema SCMS e demonstrar as vantagens oferecidas pelo modelo através de um protótipo. O protótipo desenvolvido possui três modos de operação, que permitem visualizar a importância de cada procedimento realizado pelo SCMS. No primeiro modo de operação, apresenta-se um modelo de emissão de certificados tradicional, sem privacidade, no qual a entidade emissora sabe a quem se destina o certificado. No segundo modo de operação, o sistema SCMS é aplicado parcialmente e garante-se a privacidade do usuário, mas cada requisição do veículo dá origem a um único certificado. O terceiro modo de operação melhora a eficiência deste processo. Este modo de operação representa o sistema SCMS completo e permite a obtenção de um lote arbitrariamente grande de certificados com uma única solicitação. O protótipo foi desenvolvido em Python e foi feita a interface com a biblioteca criptográfica RELIC Toolkit, escrita em C. A exibição dos dados é feita através de uma interface Web. Através do protótipo, é possível verificar a quebra de privacidade do usuário no primeiro modo de operação, pois a entidade pode associar todos os pseudônimos ao seu dono. Este problema é resolvido no segundo modo de operação, no qual a entidade emissora é incapaz de rastrear o veículo. No terceiro modo de operação, há uma faixa de funcionamento ótimo no qual observa-se melhora no tempo de execução, devido às vantagens proporcionadas pela emissão em lote. Fora da faixa de funcionamento ideal, tais vantagens são superadas pelos custos adicionais de comunicação, mas os benefícios de segurança e privacidade ainda tornam esta configuração atrativa. Além disto, o último modo possui a menor porcentagem do tempo de processamento no veículo, deixando-o livre para realizar outras atividades.

Palavras-Chave – Veículos autônomos. Comunicação veicular. Emissão de certificados. Certificado digital. Pseudônimo. Privacidade. Segurança. *Security Credential Management System.*

ABSTRACT

The emergence of autonomous and semi-autonomous vehicles is followed by prospects of accident reduction, shorter travel time, the increase of vehicles on road, among other improvements. However, vehicle automation alone is not enough. In order to provide all these benefits, the exchange of safety information between all users is required. To ensure data authenticity, all messages must be signed. User privacy is guaranteed by pseudonym usage. This document addresses the main requirements for certificate issuance in vehicular networks and discusses the most important studies of the field. Recent articles highlight the Security Credential Management System (SCMS) solution, which today represents the leading candidate for standardization in the United States. The objective of this work is to analyze the suitability of the SCMS system and to demonstrate the model advantages through a prototype. The developed prototype has three operation modes, which allow visualizing the importance of each procedure performed by SCMS. In the first operation mode, a traditional public key infrastructure is presented, without privacy, in which the issuing entity knows to whom the certificate is intended. In the second mode of operation, the SCMS system is applied partially and user privacy is guaranteed, but each vehicle request results into a single certificate. The third operation mode improves the efficiency of this process. This configuration represents the complete SCMS system and allows you to obtain an arbitrarily large batch of certificates with a single request. The prototype was developed in Python and was interfaced with the cryptographic library RELIC Toolkit, written in C. The data is displayed through a Web interface. Through the prototype, it is possible to verify the user's privacy breach in the first operation mode, since the entity can associate all the pseudonyms to its owner. This problem is solved in the second mode of operation, in which the issuing entity is unable to track the vehicle. In the third operation mode, there is an optimum operating range in which run time is improved, due to the advantages offered by the batch emission. Outside the ideal operating range, such advantages are outweighed by the additional costs of communication. Nonetheless, the security and privacy benefits still make this configuration attractive. In addition, the latter mode has the smallest percentage of vehicle processing time, which allows the device to perform other activities.

Keywords – Autonomous vehicles. Vehicle communication. Certificate issuance. Digital certificate. Pseudonym. Privacy. Safety. Security Credential Management System.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE CÓDIGOS

LISTA DE ABREVIATURAS E SIGLAS

AES *Advanced Encryption Standard*. 21, 35, 57

API *Application Program Interface*. 69

BCAM *Binary Hash Tree Based Certificate Access Management*. 31, 36

BSM *Basic Safety Message*. 15, 25

CA *Certificate Authority*. 24

DH Diffie Hellman. 23

DSRC *Dedicated Short-Range Communications*. 26

ECDSA *Elliptic Curve Digital Signature Algorithm*. 23, 54, 55, 57

ECIES *Elliptic Curve Integrated Encryption Scheme*. 23, 54, 55, 57

HMAC *Hashed Message Authentication Code*. 29

IBC *Identity-Based Cryptography*. 27

LA *Linkage Authority*. 32, 35, 37, 40, 43, 44, 47, 57, 58, 59, 60

LOP *Location Obscurer Proxy*. 69

MA *Misbehaviour Authority*. 32, 35, 69

NIST *National Institute of Standards and Technology*. 20

OSR *Order of Self-Revocation*. 30

PCA *Pseudonym Certificate Authority*. 32, 34, 35, 37, 38, 40, 43, 44, 47, 50, 54, 55, 59, 60

PKI *Public Key Infrastructure*. 16, 17, 24, 25, 26, 27, 29, 30, 31, 37, 45, 68

PUCA *Pseudonym Scheme With User-Controlled Anonymity*. 30

RA *Registration Authority*. 32, 34, 35, 37, 38, 40, 43, 44, 47, 55, 57, 58, 59, 60, 69

RSA Rivest Shamir Adleman. 23

RSU *Roadside Unit*. 28, 29

SCMS *Security Credential Management System*. 15, 16, 17, 26, 31, 32, 33, 35, 36, 37, 38, 40, 42, 43, 46, 68, 69, 70

TA *Trusted Authority*. 28

USDOT *United States Department of Transportation*. 15, 26, 31, 36

V2I *Vehicle-to-Infrastructure*. 42

V2V *Vehicle-to-Vehicle*. 26, 42

V2X *Vehicle-to-Everything*. 26, 42

SUMÁRIO

1 INTRODUÇÃO

Neste capítulo, são introduzidos os primeiros conceitos deste trabalho, como seu objetivo, sua motivação e justificativa. Também é feita uma breve descrição da organização do trabalho e separação dos capítulos.

1.1 Objetivo

O desenvolvimento de carros inteligentes e autônomos tem avançado ao longo da década de 2010 (BIMBRAW, 2015). Neste período, observou-se a criação de veículos com diferentes níveis de autonomia e que estão cada vez mais próximos de sua disponibilização como produto.

De acordo com Fagnant e Kockelman (2015), podemos esperar que os carros autônomos atuem na prevenção de acidentes, redução do tempo de viagem, aumento da eficiência de combustível, ampliação do acesso aos veículos entre diferentes faixas etárias, entre outros impactos sociais. No entanto, a atuação desses dispositivos não é isolada. É necessário o compartilhamento de informações entre os nós para formar uma visão geral da rede.

A comunicação entre os veículos deve ocorrer através da troca de mensagens básicas de segurança, ou *Basic Safety Message* (BSM) (COMMITTEE et al., 2009). A autenticidade das mensagens é garantida através da assinatura de seu remetente. Assim, cada veículo deve possuir ao menos um certificado digital emitido por entidades confiáveis e deve utilizá-lo para comprovar sua participação legítima no sistema. Com a finalidade de garantir a privacidade de seus usuários, são utilizados certificados digitais que não contêm informações sobre seus donos, chamados de pseudônimos.

Diversos estudos tentaram propor uma estrutura de comunicação com pseudônimos que fosse segura e eficiente (PETIT et al., 2015), mas é dado destaque à solução oferecida pelo Departamento de Transporte dos Estados Unidos (*United States Department of Transportation* - USDOT). O estudo *Security Credential Management System*

(SCMS)(WHYTE et al., 2013) tem ganhado destaque devido, principalmente, à clara divisão entre entidades da rede. Isso permite a geração dos certificados digitais sem que as entidades geradoras saibam para quem os certificados se destinam. Através desse método, a privacidade dos usuários é garantida mesmo quando um dos elementos da infraestrutura apresenta comportamento desvirtuado.

O objetivo deste trabalho é verificar a adequabilidade do sistema SCMS como solução para a emissão de certificados em redes veiculares. Para isso, os principais requisitos para comunicação em redes veiculares são estudados e os métodos atuais de emissão de certificados são discutidos. Após a comparação do método escolhido e outros estudos presentes no estado da arte, deseja-se demonstrar as vantagens oferecidas pela solução SCMS com relação à privacidade dos usuários e à eficiência no fornecimento de pseudônimos. Dessa forma, é desenvolvido um protótipo cujo comportamento simula a emissão de certificados e a interação entre as principais entidades do sistema. A aplicação terá três modos de operação para que diferentes mecanismos do sistema possam ser avaliados. O primeiro modo de operação aproxima-se do comportamento tradicional de uma Infraestrutura de Chaves Públicas (*Public Key Infrastructure* - PKI) e há possibilidade de rastreamento dos usuários, uma vez que não são utilizados mecanismos de privacidade. No segundo modo de operação, a solução SCMS é aplicada parcialmente, de forma que a identidade dos usuários é protegida, mas alguns mecanismos de otimização de desempenho não são utilizados. Isso torna a obtenção de certificados um processo custoso para o dispositivo. Por fim, o terceiro modo de operação representa o comportamento completo do estudo SCMS, fornecendo privacidade, segurança, escalabilidade e eficiência de processamento.

1.2 Motivação

Dada a necessidade de estabelecer uma comunicação segura entre todos os componentes da rede, o meio científico busca atualmente uma solução que atenda todos os requisitos de forma satisfatória. Apesar de ainda não existir um padrão para a comunicação veicular, a solução *Security Credential Management System* (SCMS) é um forte candidato e é utilizado como base para vários estudos.

A notoriedade garantida ao estudo SCMS origina-se da clara distinção entre as entidades de sua PKI. Assim, diferentes papéis são atribuídos a cada um de seus componentes de forma que nenhum deles seja capaz de associar um veículo com seu conjunto de chaves criptográficas. Mesmo a entidade responsável por gerar os certificados digitais não pode, isoladamente, descobrir para quem eles se destinam, pois a entidade possui informações

limitadas quanto ao veículo solicitante.

Dessa forma, a principal motivação deste trabalho é estudar o estado da arte relativo à comunicação veicular e analisar os requisitos para esse tipo de sistema, discutindo os pontos fortes e fracos de cada solução, com ênfase no modelo SCMS. Além disso, a aplicação desenvolvida busca demonstrar a superioridade da solução SCMS com relação à segurança e à privacidade do usuário ao compararmos com uma estrutura típica de PKI, na qual a entidade emissora de certificados sabe para quem eles se destinam.

1.3 Justificativa

A comunicação em redes veiculares é realizada através da troca de mensagens básicas de segurança, que contêm dados relacionados à posição do veículo, sua velocidade, a distância até o próximo carro, etc. Para prevenir a propagação de mensagens falsas pela rede, toda comunicação entre os veículos deve ser assinada. Assim, devem ser emitidos certificados para cada um dos veículos, que garantem a autenticidade dos usuários e sua participação legítima na rede.

Em um cenário no qual tais mensagens são enviadas sem mecanismos de privacidade, um atacante poderia rastrear com facilidade qualquer veículo desejado e montar um perfil sobre seus usuários. Após certo tempo de rastreio, seria possível descobrir as áreas mais visitadas por determinado dispositivo e inferir informações pessoais, como endereço de domicílio ou local de trabalho.

Em contrapartida, é desejável que o sistema seja capaz de identificar elementos mal-intencionados da rede e impôr consequências às suas ações. Um atacante poderia enviar mensagens informando condições falsas de trânsito em uma via, com o intuito de liberar sua passagem por determinado caminho. Outro possível ataque envolve o envio de mensagens falsas informando um acidente à frente, de forma que o veículo em aproximação seja obrigado a parar. Tal comportamento pode ser propício para a execução de roubos ou sequestros, colocando em risco a vida dos usuários legítimos da rede.

Redes veiculares devem garantir, portanto, a privacidade dos usuários e a irretratabilidade de suas ações. A combinação e o balanceamento desses dois requisitos são um dos principais desafios do setor e são temas discutidos ao longo deste trabalho.

1.4 Organização do Trabalho

A organização deste trabalho foi dividida em seis capítulos. O primeiro capítulo, à qual esta seção pertence, corresponde à introdução ao problema e discussão sobre sua motivação.

O segundo capítulo aborda aspectos conceituais utilizados neste trabalho. Este documento foi desenvolvido considerando a participação da aluna no programa de Pré-Mestrado de Pesquisa em Engenharia da Computação. Devido ao enfoque em pesquisa, a seção de aspectos conceituais é mais explorada do que ao compararmos com outros trabalhos de conclusão de curso. O capítulo inicia-se com a fundamentação teórica necessária para as seções seguintes, abordando conceitos como função de hash e criptografia. Em sequência, é apresentada a seção de revisão da literatura, na qual é descrito o cenário atual para a comunicação veicular. São abordados quatro tipos de soluções existentes e dá-se destaque para o estudo mais promissor.

O terceiro capítulo apresenta a especificação dos requisitos do sistema e é descrito o funcionamento geral da aplicação. São discutidos os métodos empregados e são levantados os requisitos funcionais e não funcionais do sistema, além da definição da arquitetura utilizada. O quarto capítulo discute as tecnologias escolhidas e os motivos que levaram a essas decisões.

O quinto capítulo discute o projeto e implementação do sistema. O desenvolvimento é dividido em etapas e são descritos os procedimentos utilizados para atingir cada objetivo. Este capítulo também aborda os testes realizados no sistema e seus resultados.

O sexto e último capítulo consiste nas considerações finais do projeto, a partir das quais são retiradas as conclusões. Por fim, são discutidas as contribuições do trabalho e as perspectivas de continuidade.

2 ASPECTOS CONCEITUAIS

Os aspectos conceituais permitem a compreensão do projeto desenvolvido. Na primeira seção, são discutidos assuntos básicos de segurança, cujo conteúdo foi abordado ao longo do curso de graduação. Na segunda seção, são abordados artigos pertencentes ao estado da arte de comunicação veicular e emissão de certificados.

2.1 Fundamentação teórica

Este capítulo aborda os conceitos básicos para a compreensão deste trabalho. São discutidos mecanismos como função de hash, criptografia simétrica, assimétrica e certificado digital. Caso o leitor já esteja familiarizado com tais assuntos, é possível avançar para o capítulo 2.2.

2.1.1 Funções de Hash

Uma função de hash aceita uma mensagem M de tamanho variado como entrada e produz resumo de tamanho fixo $h = H(M)$ como saída (STALLINGS et al., 2012). A principal vantagem de seu uso está relacionada à integridade dos dados, uma vez que a alteração de um bit na mensagem de entrada produzirá uma mensagem de saída completamente diferente.

Funções de hash podem ser aplicadas a blocos de qualquer tamanho e devem produzir blocos de saída de tamanho fixo. O cálculo do valor de hash não depende de nenhuma chave secreta, sendo necessária apenas a mensagem de entrada. Além disso, funções de hash devem ser resistentes a três tipos de ataque:

- Resistência à primeira inversão: Para um resumo h, deve ser computacionalmente impossível encontrar x tal que $H(x) = h$.
- Resistência à segunda inversão: Dado um bloco de entrada x, deve ser computacio-

nalmente impossível encontrar um bloco $y \neq x$ tal que $H(y) = H(x)$.

- Resistência a colisões: Deve ser computacionalmente impossível encontrar um par (x, y) tal que $H(x) = H(y)$.

Funções de hash são amplamente utilizadas em contextos que requerem integridade, como certificados digitais, *download* de arquivos na Internet, etc. Elas também podem ser aplicadas no armazenamento seguro de senhas em bancos de dados através de uma técnica conhecida como *password hashing*.

Os algoritmos recomendados para uso atualmente são SHA-224, SHA-256, SHA-384, SHA512, todos pertencentes à família SHA-2 (DANG, 2015). O tamanho dos resumos produzidos corresponde ao número presente no nome dos algoritmos (e.g., SHA-256 produz resumos com 256 bits). O algoritmo antecessor à família SHA-2, o SHA-1, não é mais recomendado para uso em aplicações futuras, uma vez que pesquisadores foram capazes de descrever um ataque de colisão no algoritmo (WANG; YIN; YU, 2005). Por fim, uma competição de algoritmos de hash criptográficos realizada entre 2008 e 2012 pela *National Institute of Standards and Technology* (NIST) elegeu o algoritmo KECCAK (BERTONI et al., 2009) como o novo sucessor SHA-3 (DWORKIN, 2015).

2.1.2 Criptografia simétrica

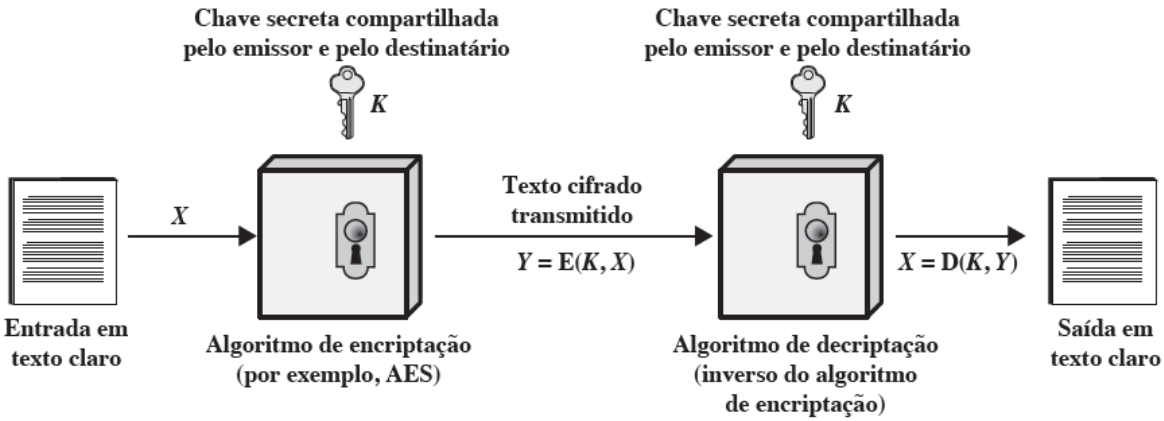
Algoritmos criptográficos de chaves simétricas, também conhecidos como cifras de chave única, têm como objetivo a cifração de uma mensagem a partir de uma única chave compartilhada, que deve ser mantida secreta por ambos os lados da comunicação (STALLINGS et al., 2012). Dessa forma, a cifra simétrica é uma transformação matemática inversível aplicada a uma mensagem, cujo processo de cifração e decifração dependem apenas de uma única chave secreta.

De acordo com Stallings (2003), um esquema de encriptação simétrica possui cinco itens:

- Texto claro: é a mensagem original, cujos dados estão às claras e desejamos cifrar.
- Algoritmo de encriptação: realiza transformações e substituições no texto claro.
- Chave secreta: É utilizada no algoritmo de encriptação para realizar as transformações e substituições, de acordo com o detalhes do algoritmo. A utilização de uma chave diferente para cifrar a mesma mensagem produzirá uma saída diferente.

- Texto cifrado: É a saída do algoritmo e depende do valor do texto claro e da chave secreta. O texto cifrado possui aparência aleatória e deve ser ininteligível.
- Algoritmo de decifração: Transformação aplicada sobre o texto cifrado em conjunto com a chave secreta, que produz como saída o texto original às claras. Consiste, basicamente, no algoritmo de encriptação executado no modo inverso.

Figura 1: Esquema simplificado de criptografia simétrica



Fonte: (STALLINGS, 2003)

Dentre as soluções existentes, o algoritmo *Advanced Encryption Standard* (AES) (DAEMEN; RIJMEN, 1999) corresponde a um dos mais populares, sendo utilizado para diversas aplicações, como sistemas seguros de comunicação, banco de dados de alto desempenho, *smart cards*, etc. O AES é uma cifra de bloco com chaves de 128, 192 ou 256 bits e blocos de tamanho fixo contendo 128 bits. A cifração de mensagens maiores depende da utilização de um modo de operação, como, por exemplo, o modo *cipher-block chaining* (CBC).

2.1.3 Criptografia assimétrica

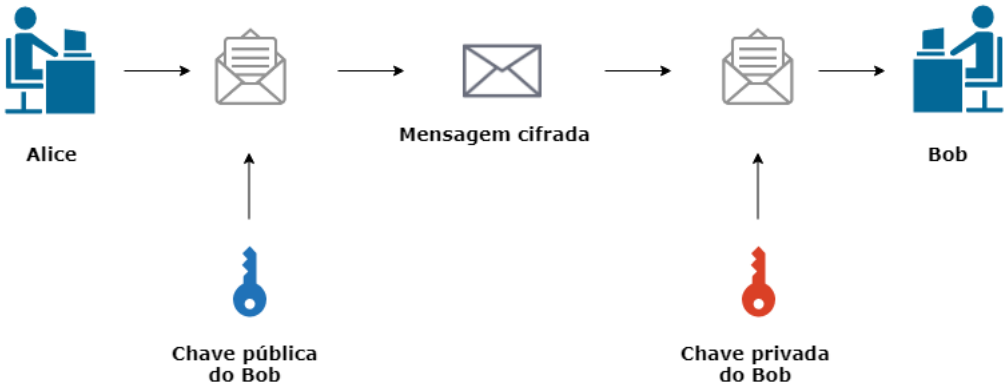
A criptografia de chave pública, ou criptografia assimétrica, baseia-se na utilização de um par de chaves, uma pública e outra privada, para realizar operações complementares, como cifração e decifração ou geração e verificação de assinatura (WILLIAM, 2006). Os algoritmos de criptografia assimétrica baseiam-se em problemas matemáticos muito difíceis de serem resolvidos, mas fáceis de serem verificados, como fatoração de números inteiros, logaritmo discreto, etc. Além disso, deve ser computacionalmente inviável de-

terminar a chave privada a partir de sua chave pública, mesmo com amostras do texto cifrado disponíveis.

Nos algoritmos assimétricos, ambas as chaves devem ser geradas pelo seu dono. A chave privada deve ser guardada e mantida em segredo, enquanto que a chave pública pode ser divulgada e distribuída para os usuários desejados. Tal característica é necessária pois as operações realizadas com uma das chaves podem ser anuladas pela outra.

O processo de cifração de mensagens pode ser visualizado na figura 2. Alice deseja enviar uma mensagem para outro usuário, Bob, e ela deseja que apenas Bob seja capaz de visualizar o conteúdo da mensagem. Alice utiliza a chave pública de Bob, conhecida por todos da rede, para cifrar a mensagem. A única pessoa capaz de decifrar esta mensagem é Bob, que possui a chave privada.

Figura 2: Cifração de mensagens através de criptografia assimétrica

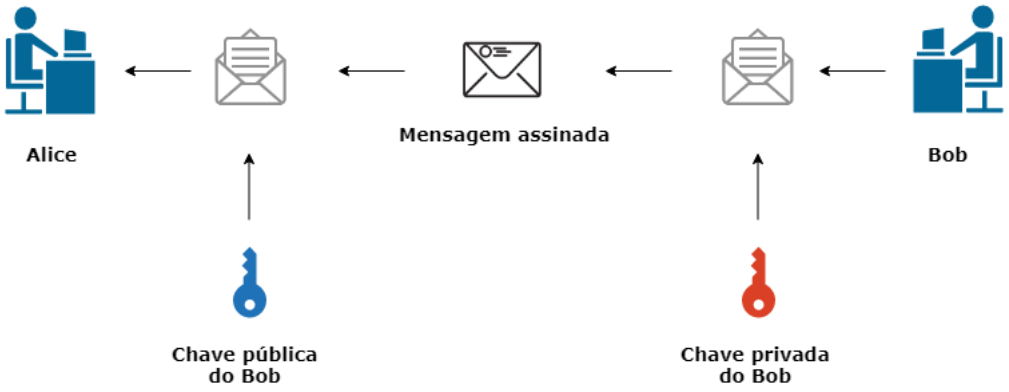


Fonte: Autoria própria

Para assinaturas, a chave privada é responsável por assinar a mensagem, enquanto que a chave pública permite a sua verificação. Caso Bob deseje enviar um documento assinado para Alice, de forma que todos possam verificar sua autoria, Bob deve assinar a mensagem utilizando sua chave privada. Alice, e qualquer outro usuário interessado, pode verificar a mensagem empregando a chave pública de Bob. Esse comportamento pode ser visualizado na figura 3. Através da verificação correta da mensagem é inferido que apenas Bob poderia tê-la assinado, o que confere irretratabilidade à operação.

A criptografia assimétrica pode ser utilizada também para a distribuição de chaves entre usuários, que é a principal dificuldade relacionada ao uso de algoritmos simétricos. Apesar de resolver o problema de distribuição de chaves, algoritmos assimétricos são muito custosos computacionalmente ao compararmos com a execução de algoritmos simétricos. Devido ao custo adicional de processamento, algoritmos assimétricos são frequentemente

Figura 3: Assinatura de mensagens através de criptografia assimétrica



Fonte: Autoria própria

utilizados apenas para o estabelecimento de chaves. Uma vez definidas, prossegue-se a comunicação por meio de algoritmos simétricos.

O algoritmo assimétrico mais utilizado atualmente é o Rivest Shamir Adleman (RSA), cuja denominação é derivada dos sobrenomes de seus criadores (RIVEST; SHAMIR; ADLEMAN, 1978). O RSA pode ser utilizado tanto para assinatura quanto para cifração e seu funcionamento baseia-se na dificuldade de fatorar números inteiros grandes. Suas chaves devem ter tamanho mínimo de 3072 bits para garantir segurança da ordem de 128 bits.

O principal problema com o algoritmo RSA reside na necessidade de chaves muito grandes para garantir um nível de segurança aceitável. Nesse contexto, surgem algoritmos baseados em curvas elípticas, que utilizam chaves de tamanho mais reduzido. Nesse tipo de algoritmo, a chave privada é representada por um número inteiro b aleatório, normalmente representado neste documento por uma letra minúscula. A chave pública corresponde a um ponto P da curva elíptica escolhida, normalmente representado neste documento por uma letra maiúscula. O valor de P deve obedecer a $P = b * G$, no qual o operador $*$ representa a multiplicação do inteiro por um ponto G e G corresponde a um ponto fixo gerador da curva, conhecido por todos. O funcionamento desses algoritmos baseia-se na dificuldade de descobrir o valor b ao conhecer os valores P e G. Os procedimentos matemáticos por trás dos algoritmos não são necessários para a compreensão deste documento, porém mais informações podem ser encontradas no livro de Hankerson, Menezes e Vanstone (2006).

Dentre os algoritmos baseados em curvas elípticas, o algoritmo *Elliptic Curve Digital Signature Algorithm* (ECDSA) é utilizado para assinaturas e o algoritmo *Elliptic Curve*

Integrated Encryption Scheme (ECIES) é utilizado para cifração. Ambos utilizam chaves de 256 bits para fornecer segurança de 128 bits. A utilização de parâmetros menores, em comparação com RSA ou DH, torna as computações mais rápidas para esses algoritmos, além de diminuir o tamanho dos certificados gerados.

2.1.4 Certificado digital

Certificados digitais são documentos assinados digitalmente que permitem a verificação de seu dono e a prova de autoria para terceiros. Através de certificados digitais, é possível conferir alguns serviços básicos de segurança, como autenticidade, irretratabilidade e integridade.

Ao considerarmos textos grandes, assiná-los diretamente dobraria o tamanho da mensagem transmitida, o que pode tornar-se inviável à medida em que os textos crescem. A solução utilizada atualmente baseia-se no uso de funções de hash para gerar a assinatura. O texto é submetido à uma função de hash e o resumo resultante é assinado com a chave privada de seu dono. O texto às claras é transmitido juntamente com o resumo assinado, permitindo que qualquer um com acesso à sua chave pública verifique a validade do documento. Esse tipo de procedimento também garante a integridade do texto original, uma vez que alterações no bloco de entrada modificariam o valor de seu resumo e este não corresponderia ao resumo assinado.

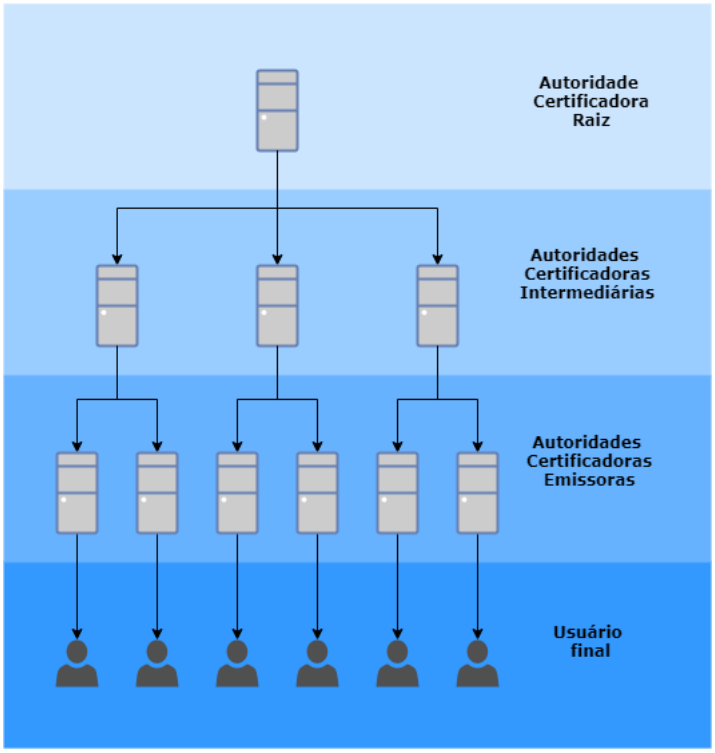
Para que os outros usuários possam realizar a verificação, é necessário que a chave pública seja enviada juntamente com o documento e seu resumo assinado. Isso cria a possibilidade de atacantes forjarem a mensagem ao alterar o documento original, já que seria necessário apenas gerar um hash e assiná-lo com uma chave privada própria. A chave pública falsa seria adicionada ao certificado e outros usuários da rede não desconfiariam que o autor da mensagem não é quem diz ser.

A prevenção desse tipo de ataque é feita com a presença de uma Autoridade Certificadora (*Certificate Authority* - CA), que é responsável por validar a chave pública de um usuário, atestando que, de fato, ela lhe pertence. A função desempenhada pela CA é assinar o certificado digital do usuário, que deve conter suas informações básicas (e.g., nome e CPF) e sua chave pública.

Para garantir a autenticidade da chave pública da CA, precisamos de uma infraestrutura de chaves públicas (*Public Key Infrastructure* - PKI). O objetivo principal para desenvolver uma PKI é permitir a aquisição segura, conveniente e eficiente de chaves

públicas (WILLIAM, 2006). Seu funcionamento baseia-se em uma autoridade certificadora central confiável, como exibido na figura 4. Essa autoridade raiz emitirá certificados para autoridades certificadoras intermediárias, que, por sua vez, emitirão para autoridades certificadoras emissoras. Essas devem emitr certificados para o usuário final, criando assim uma cadeia de confiança entre autoridades.

Figura 4: Estrutura hierárquia de uma PKI



Fonte: Autoria própria

Existem outros mecanismos de distribuição de chaves, mas a utilização de uma PKI é a mais comum para assinaturas digitais. Caso seja necessário revogar o certificado de um usuário, deve-se adicioná-lo a uma lista de certificados revogados, que pode funcionar conectada, ou não, à rede.

2.2 Revisão da literatura

A comunicação veicular tem como objetivo conscientizar usuários das condições atuais de uma via, podendo ser implementada tanto em veículos autônomos quanto em semi-autônomos. Através do envio de mensagens básicas de segurança (*Basic Safety Message* - BSM), os veículos informam sua posição, velocidade, distância em relação a outros carros, além de poder fornecer dados relativos ao trânsito e a acidentes próximos de sua

localização. O envio e recebimento de tais mensagens permite mapear o estado atual de uma via, possibilitando a sugestão de novos caminhos.

A existência de veículos maliciosos, no entanto, pode produzir conclusões errôneas sobre o trajeto atual do usuário. Caso sejam enviadas mensagens falsas informando um acidente à frente do usuário, o veículo pode ser obrigado a reduzir sua velocidade ou parar completamente. Nesse momento, poderia ser organizado um assalto ou sequestro aos ocupantes do veículo, colocando em risco sua segurança.

Outro cenário, menos perigoso, refere-se ao envio de mensagens falsas informando o congestionamento de uma via. Nesse caso, os veículos receptores de tal mensagem alterariam seu trajeto original, com o objetivo de evitar tal via, e isso permitiria a passagem livre do veículo malicioso.

A ocorrência de tais eventos deve ser mitigada através de um mecanismo de autenticação dentro da rede, como, por exemplo, a assinatura das mensagens enviadas. Isso permite verificar que determinado usuário pertence à rede e está apto a transmitir informações. Caso ele apresente comportamento desvirtuado, sua inscrição deve ser revogada e sua participação suspensa.

Hoje, é possível encontrar no mercado automóveis semi-autônomos que possuam sistema de comunicação veicular. Em 2017, a marca Cadillac lançou modelos de CTS Sedans que incluíam sistemas de comunicação V2V. Os modelos utilizam a tecnologia de comunicação dedicada de curto alcance (*Dedicated Short-Range Communications* - DSRC) e podem emitir 1000 mensagens por segundo para veículos a pouco mais de 300 metros de distância (CADILLAC, 2017). No mesmo ano, a marca Mercedes-Benz (MERCEDES-BENZ, 2017) lançou Sedans Classe-E capazes de atuar na comunicação Car-to-X, ou seja, entre carros e "tudo", também conhecido como V2X. As divergências entre as implementações já podem ser observadas na forma de comunicação escolhida, uma vez que os modelos da Mercedes-Benz utilizam redes de telefonia móvel para a transmissão de mensagens, ao invés de tecnologias de curto alcance. O protocolo de comunicação escolhido permite apenas a troca de mensagens com veículos do mesmo modelo, o que limita as informações compartilhadas. Outras companhias já anunciaram a inclusão de modelos com comunicação V2X, com a previsão da Volkswagen para 2019 (EDELSTEIN, 2017) e a previsão da Toyota para 2021 (ABUELSAMID, 2018).

Apesar de já existirem soluções disponíveis no mercado, não há consenso entre a comunicação dos veículos de diferentes marcas. Com relação ao protocolo e emissão de certificados, ainda não foi estabelecido um padrão de gerenciamento da rede, seja através

de uma PKI ou outro mecanismo. O rápido desenvolvimento do setor de carros inteligentes tem contribuído na aceleração da pesquisa relativa à comunicação veicular, uma vez que deseja-se uniformização para aplicações com destino ao mercado. A solução SCMS é forte candidata para padronização nos Estados Unidos pelo USDOT e é discutida com mais detalhes na seção 2.2.2.

2.2.1 Cenário atual na comunicação veicular

A utilização de um único certificado de identificação em uma rede veicular pode permitir o rastreamento do veículo, tanto por outros usuários quanto por entidades da própria rede (SCHAUB; MA; KARGL, 2009). Caso um usuário sempre assine suas mensagens com o mesmo certificado, é fácil para qualquer membro da rede acompanhar seu trajeto através das mensagens enviadas e descobrir informações valiosas sobre seus passageiros, como o local de sua moradia, de seu trabalho, etc. Uma solução para este problema consiste na emissão de um conjunto de certificados, chamados de pseudônimos, que não contêm informações sobre seu usuário. Os pseudônimos permitem a autenticação do veículo sem identificar seu dono, uma vez que a obtenção de pseudônimos depende da participação válida do usuário na rede.

Dentre os estudos realizados na área de comunicação veicular e emissão de certificados, Petit et al. (2015) classificam as soluções atuais em quatro categorias: soluções com pseudônimos baseadas em criptografia assimétrica e PKI; soluções com criptografia baseada em identidade; soluções de assinatura em grupo; e soluções baseadas em criptografia simétrica. Nas subseções 2.2.1.1, 2.2.1.2, 2.2.1.3 e 2.2.1.4 serão discutidos os mecanismos utilizados em cada uma das categorias e alguns estudos correspondentes.

2.2.1.1 Soluções com criptografia baseada em identidade

Criptografia baseada em identidade, ou *Identity-Based Cryptography* (IBC)(SHAMIR, 1984), tem como premissa a utilização da própria identidade do usuário como sua chave pública. A partir da definição de um identificador único (e.g., um endereço de e-mail, um identificador de rede, etc.), o usuário obtém sua chave privada correspondente para ser utilizada na comunicação. Para prevenir a derivação de chaves privadas a partir da identidade de outros usuários, este processo somente pode ser realizado por uma entidade confiável com conhecimento de todos os parâmetros da rede. Este modelo apresenta características de autenticação similares às do modelo de PKI, sem a necessidade, no entanto, de armazenar chaves públicas e trocar certificados. A desvantagem é a necessidade

de possuir uma entidade confiável, que conheça todas as chaves privadas dos veículos da rede. A utilização de pseudônimos pode ser aplicada a este modelo através da utilização de trechos aleatórios de texto como chave pública, ao invés de identificadores do usuário.

Kamat, Baliga e Trappe (2006) propõem a decentralização da emissão de certificados, uma vez que este trabalho é realizado apenas pela entidade confiável (*Trusted Authority* - TA). Nesse cenário, a emissão do certificado de identificação principal do veículo é realizada pela TA, que publica os parâmetros do sistema para unidades pertencentes à infraestrutura das estradas (*Roadside Unit* - RSU). As RSU recebem tais parâmetros e são responsáveis pela emissão dos pseudônimos, reduzindo o processamento feito pela TA.

Outra solução, proposta por He et al. (2015), atribui ao próprio veículo a geração dos pseudônimos, através do uso de um hardware inviolável. A TA é responsável por gerar um valor RID correspondente à identidade real do veículo e uma senha PWD para acesso ao hardware de segurança, que são configurados ao início da vida útil do veículo. O veículo solicita certificados anônimos ao seu hardware de segurança, que irá verificar se os valores RID e PWD fornecidos pelo usuário são correspondentes à sua configuração. Caso afirmativo, o pseudônimo é gerado e entregue ao sistema do veículo.

2.2.1.2 Soluções com assinatura em grupo

Chaum e Heyst (1991) propuseram um novo modelo de assinatura, conhecido como assinatura em grupo. Este modelo deve atender a três características: *a*) somente membros do grupo podem assinar mensagens, *b*) o receptor da mensagem pode verificar que a assinatura é válida, mas não consegue identificar qual dos membros do grupo a fez, e *c*) se necessário, a identidade de quem assinou a mensagem pode ser revelada por uma autoridade da rede.

Cada grupo possui uma chave pública compartilhada, que pode ser utilizada por qualquer um de seus membros. Cada membro, por sua vez, possui sua própria chave privada, fornecida pelo gerenciador do grupo. As operações realizadas com as chaves privadas podem ser desfeitas utilizando a mesma chave pública do grupo. Isso garante anonimidade e autenticidade a seus membros, uma vez que a assinatura não revela a autoria da mensagem, mas confirma a participação do usuário no grupo.

Os problemas relacionados a esse tipo de abordagem referem-se à confiança depositada no gerenciador do grupo, uma vez que ele tem acesso a todas as chaves privadas e poderia expor a identidade dos usuários. Além disso, a revogação de certificados é problemática, já que a suspensão de um único membro implica na obtenção de uma nova

chave pública e, consequentemente, obriga todos os outros usuários a obter novas chaves privadas. Essa limitação reduz a escalabilidade do sistema ao permitir apenas pequenos grupos de operação, o que pode representar um gargalo no desempenho do sistema.

A abordagem proposta por Calandriello et al. (2007) caracteriza um comportamento híbrido entre duas das classificações propostas. A utilização de uma PKI tradicional é combinada com assinatura em grupo para a geração de pseudônimos. Assim, cada grupo é equipado com uma chave privada única e uma chave pública correspondente que é comum a um grupo de veículos. Ao invés de utilizar essas chaves para assinar e verificar as mensagens, o usuário deve gerar seus próprios pseudônimos e assiná-los com sua chave privada de grupo. Isso permite que qualquer elemento pertencente ao grupo possa verificar a autenticidade da assinatura através da chave comum pública. Através dessa solução, o problema de reabastecimento de pseudônimos é resolvido, uma vez que esses são gerados pelo próprio veículo. A utilização de assinatura em grupo, no entanto, é acompanhada dos problemas associados a essa categoria, principalmente na revogação de certificados e modificação dos grupos.

Zhang et al. (2010) também utilizaram assinaturas em grupo para garantir a comunicação segura entre os veículos. Sua proposta é a utilização das RSU para gerenciar os grupos, ao invés de uma autoridade centralizada. Cada RSU seria responsável por manter e gerenciar um grupo de veículos dentro do seu alcance de comunicação. Ao dirigirem por uma RSU, os veículos poderiam solicitar uma nova chave privada para participar do grupo ou então para renovar sua chave privada expirada. A revogação dos pseudônimo poderia ser feita por uma terceira entidade independente.

2.2.1.3 Soluções baseadas em criptografia simétrica

As soluções baseadas em criptografia simétrica possuem fortes limitações na sua operação, mas garantem um bom desempenho computacional, dada a rapidez das cifras simétricas. Assim, a menção de sua existência é válida, mesmo que as restrições dificultem sua aplicação prática.

Seu funcionamento baseia-se no uso de códigos de autenticação (*Hashed Message Authentication Code* - HMAC), no qual uma função de hash é aplicada sobre a mensagem e uma chave secreta. Para fazer a verificação da mensagem, outros usuários devem ter conhecimento da mesma chave secreta, o que limita a anonimidade do veículo ao grupo conhecedor da chave. Esta solução sozinha não garante irretratabilidade, uma vez que não é possível identificar quem enviou cada mensagem. Dessa forma, algumas características

de criptografia assimétrica devem ser emuladas para fornecer não-repúdio à comunicação.

Como exemplo, é abordada a solução de Perrig et al. (2005), que utiliza como base a sincronização fraca (*loosely time synchronized*) entre os participantes da conversa. O transmissor da mensagem divide o tempo em intervalos de duração uniforme. Para cada intervalo, é utilizada uma chave simétrica ligada diretamente com as seguintes, através de um mecanismo conhecido como *one-way chain*. Uma função f irreversível (e.g., hash) é aplicada sobre uma chave x_0, gerando uma nova chave x_1. A operação é repetida até obter-se x_n, para o valor de n desejado. Quando a corrente estiver pronta, seus nós são utilizados na ordem reversa, iniciando com x_n e retornando até atingir x_0. A solução de Perrig et al. propõe o envio de uma mensagem M_1 autenticada com uma chave K_1 no tempo $t = 1$. O veículo que receber a mensagem não consegue verificá-la ainda e, portanto, deve armazená-la. Em um momento $t = 2$, o veículo transmissor envia o valor da chave K_1 juntamente com uma nova mensagem autenticada com K_2. Neste momento, o veículo receptor pode verificar a primeira mensagem, enquanto a segunda é armazenada em seu dispositivo. A principal desvantagem desse modelo é o atraso na recepção e verificação das mensagens, o que pode gerar problemas de segurança para mensagens de teor crítico.

2.2.1.4 Soluções baseadas em criptografia assimétrica

Soluções baseadas em criptografia assimétrica têm apostado no uso de pseudônimos. As mensagens são assinadas com a chave privada do pseudônimo e enviadas juntamente com seu certificado, de forma a facilitar sua verificação. As principais dificuldades relacionadas ao uso de pseudônimos em um modelo de PKI residem na troca, reabastecimento e proteção da privacidade dos pseudônimos contra entidades emissoras mal-intencionadas. Quando ocorre a expiração de um pseudônimo, o veículo deve carregar outro certificado de seu armazenamento interno ou solicitar novos a uma autoridade certificadora.

A premissa principal do estudo *Pseudonym Scheme With User-Controlled Anonymity* (PUCA) (FÖRSTER; KARGL; LÖHR, 2014) é garantir a anonimidade dos usuários contra atacantes e entidades maliciosas da rede. Cada veículo deve gerenciar seus certificados de longa duração, assim como pseudônimos, utilizando um módulo de confiança em hardware capaz de suspender sua própria operação. Assim, caso o usuário apresente comportamento fora dos padrões estabelecidos, este receberá uma ordem de autorrevogação (*Order of Self-Revocation* - OSR) destinada a um de seus pseudônimos, o que desencadeará a suspensão de operação do módulo e a auto-revogação do veículo. No cenário proposto pelos autores, não é necessário revelar a identidade dos pseudônimos e a privacidade dos

usuários nunca é comprometida. Além disso, os pseudônimos não são ligados uns aos outros, impedindo a descoberta do lote a partir de um único certificado. A solução de revogação, no entanto, é propensa à má-conduta do usuário, uma vez que o veículo poderia tentar evitar mensagens OSR ou modificar seu módulo de confiança para não suspender a operação.

A solução *Binary Hash Tree Based Certificate Access Management* (BCAM) (KUMAR; PETIT; WHYTE, 2017) utiliza uma abordagem diferente quanto ao provisionamento de pseudônimos e sua utilização. Logo na configuração do dispositivo, o veículo recebe todos pseudônimos referentes à sua vida útil. Esses pseudônimos, no entanto, não podem ser imediatamente utilizados, uma vez que os certificados estão cifrados, juntamente com os valores para a reconstrução das chaves privadas. Quando um lote de certificados aproxima-se de sua data de vigência, o veículo recebe um código para a decifração do lote, permitindo o cálculo das chaves privadas e a utilização dos pseudônimos. Caso um veículo seja revogado por má-conduta, a entidade de ativação dos códigos não deve fornecer as chaves de decifração para o veículo suspenso, impedindo, assim, sua participação na rede.

Por fim, outra solução que utiliza o modelo PKI e criptografia assimétrica é o estudo SCMS. O comportamento do sistema é detalhado na seção 2.2.2.

2.2.2 *Security Credential Management System*

Dentre as soluções propostas para comunicação veicular, o estudo *Security Credential Management System* (WHYTE et al., 2013) têm recebido grande destaque no meio acadêmico, sendo considerado um dos principais candidatos para padronização nos Estados Unidos pelo USDOT. A proposta do SCMS é o desenvolvimento de uma PKI específica para comunicação veicular, na qual aspectos de privacidade e segurança são levados em consideração. Outro diferencial do SCMS em relação a modelos de PKI tradicionais é o foco em eficiência e escalabilidade.

O sistema busca prevenir-se contra ataques externos e internos à privacidade do usuário. Os ataques externos são combatidos com a troca constante de certificados Os veículos são equipados com dois tipos de certificados: um certificado de inscrição, que possui longa duração e é utilizado para identificar o veículo; e pseudônimos, que devem durar apenas alguns dias e são emitidos em grande quantidade. Durante a inicialização do sistema, o veículo é equipado com uma grande quantidade de pseudônimos, equivalente a até 3 anos de uso total. Durante um período de tempo t, $\sigma \geq 1$ pseudônimos são válidos,

de forma que o veículo possa alternar seu uso ao assinar mensagens. No entanto, o valor de σ deve ser limitado para evitar ataques do tipo *Sybil* (DOUCEUR, 2002). WHYTE et al. estabelecem o período de uso de pseudônimos como uma semana, sendo que cada lote contém entre 20 a 40 certificados válidos simultaneamente.

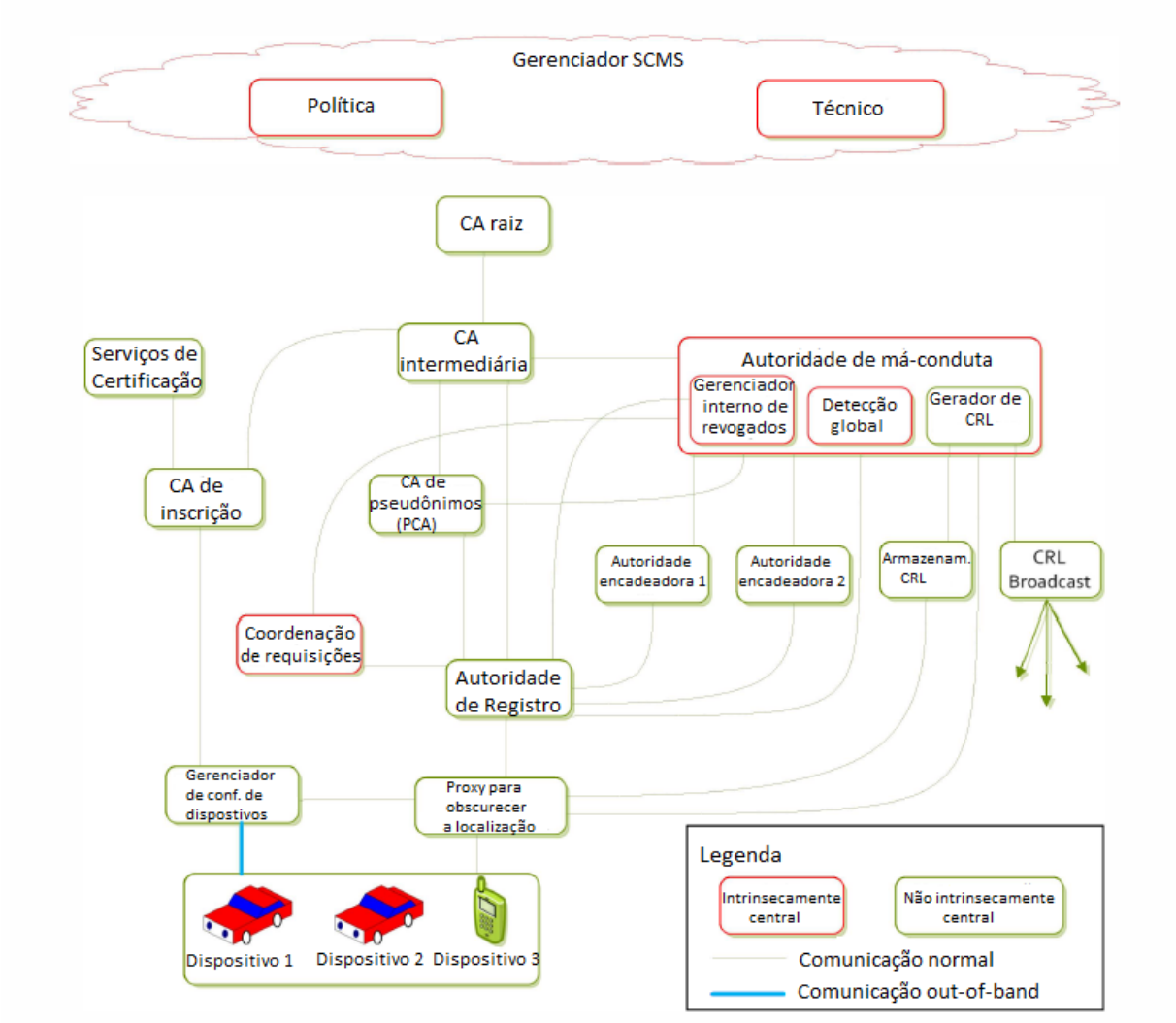
Os ataques internos são realizados por entidades da própria rede, que podem apresentar comportamento do tipo malicioso ou do tipo "honesto, mas curioso". O último comportamento é referente a entidades que desempenham suas ações corretamente, mas que podem executar ataques passivos contra o sistema, como utilizar ou divulgar dados do usuário a que tenham acesso. O sistema previne-se contra esse tipo de ataque através da clara divisão entre papéis das entidades e da restrição de informações recebidas por cada uma. Dessa forma, nenhuma entidade conhece o suficiente sobre o veículo para rastreá-lo.

A arquitetura do SCMS é apresentada na figura 5. As principais entidades do sistema e suas funções são:

- Dispositivo: Corresponde ao próprio veículo.
- Autoridade certificadora de pseudônimos (*Pseudonym Certificate Authority* - PCA): responsável por emitir os pseudônimos para os veículos, a pedido da RA.
- Autoridade registradora (*Registration Authority* - RA): recebe as solicitações de pseudônimos dos dispositivos, assinados com seu certificado de inscrição. Após a validação do certificado, faz a expansão das *butterfly keys*, dando origem a múltiplas solicitações de pseudônimos. Embaralha as requisições de vários dispositivos e as envia para PCA.
- Autoridade encadeadora (*Linkage Authority* - LA): gera uma semente aleatória, que é expandida em múltiplas sequências pseudoaleatórias de bits. Essas sequências são inseridas nos pseudônimos e permitem seu encadeamento e revogação em lote, dado que pertençam ao mesmo dono. No SCMS, são utilizadas duas entidades LA.
- Autoridade de má-conduta (*Misbehaviour Authority* - MA): responsável por monitorar a rede e avaliar denúncias de mau comportamento entre os usuários. Caso seja necessário, a MA é responsável por revogar os certificados do veículo, banindo-o da comunicação.

A emissão e revogação eficiente dos certificados dependem de dois procedimentos importantes: a expansão das chaves borboletas (*butterfly key expansion*) e seu encadeamento.

Figura 5: Arquitetura geral do sistema SCMS.



Fonte: Adaptado de (WHYTE et al., 2013)

A expansão é iniciada pelo veículo e realizada pelas entidades RA e PCA, enquanto que o encadeamento é realizado, principalmente, por duas LA, com participação da PCA e RA. O encadeamento pode ser utilizado pela MA para a revogação dos certificados.

2.2.2.1 Expansão de chaves borboletas

A expansão de chaves permite a obtenção de um lote arbitrariamente grande de pseudônimos a partir de uma única requisição. Pseudônimos são assinados e encriptados individualmente utilizando uma chave única para cada certificado. O estudo SCMS utiliza criptografia assimétrica baseada em curvas elípticas para a demonstração da expansão de chaves.

O veículo gera dois pares de chaves, conhecidas como chaves casulo ou *cocoon keys*. Cada par de chaves corresponde a um inteiro a (chave privada) e um ponto $A = aG$ (chave pública), com G representando um ponto base previamente definido pelas entidades e conhecido por todos. O veículo também deve gerar duas funções f_1 e f_2, que correspondem a uma permutação pseudo-aleatória de inteiros.

As chaves públicas e as funções são transmitidas cifradas para a RA. Cada chave, em conjunto com uma das funções, é utilizada para obter um lote com $n > 0$ novas chaves públicas, conhecidas como chaves lagarta ou *caterpillar keys*. A expansão das chaves lagarta gera chaves públicas através da operação $B_i = A + f_k(i) * G$, com $i >= 0$ e $i < n$. Por sua vez, as chaves privadas são geradas através do cálculo $b_i = a + f_k(i)$. O veículo é o único que conhece ambos os valores de a e $f_k(i)$ e, portanto, é o único que pode calcular o valor das chaves privadas. O envio de duas chaves públicas por parte do veículo dá origem a dois lotes de expansões: um dos lotes, cujas chaves são referenciadas como B_i, dá origem aos pseudônimos finais, enquanto que o outro lote, cujas chaves são referenciadas como J_i é utilizado para a cifração dos certificados.

Após a expansão, a RA é responsável por montar as requisições e enviá-las à PCA. Cada requisição contém uma chave B_i e uma chave J_i, o que totaliza n requisições para o mesmo veículo. A RA acumula um grande número de requisições de diferentes veículos, que são embaralhadas antes de seu envio à PCA. Este procedimento impede a PCA de descobrir a quem o certificado se destina, ou se dois certificados pertencem ao mesmo veículo.

A geração dos pseudônimos é feita pela PCA. A chave B_i recebida é expandida em uma nova chave pública, conhecida como chave borboleta ou *butterfly key*. A PCA gera um inteiro aleatório c e calcula um ponto $C = cG$. A chave que é incluída no certificado final é o valor $B_i + C$. De forma que o pacote não seja substituído, a PCA deve assiná-lo como um todo, assegurando assim sua autoria. Em seguida, o certificado é cifrado utilizando a chave J_i recebida na requisição. Isso impede a RA de descobrir o valor dos pseudônimos e garante a privacidade do usuário. O pacote final contém o certificado e o valor c aleatório utilizado na expansão.

Através da RA, a PCA retorna para o veículo o pacote cifrado e assinado. O veículo deve reconstruir as chaves privadas dos pseudônimos, utilizando as informações recebidas da PCA e suas chaves privadas casulo originais.

2.2.2.2 Encadeamento de certificados

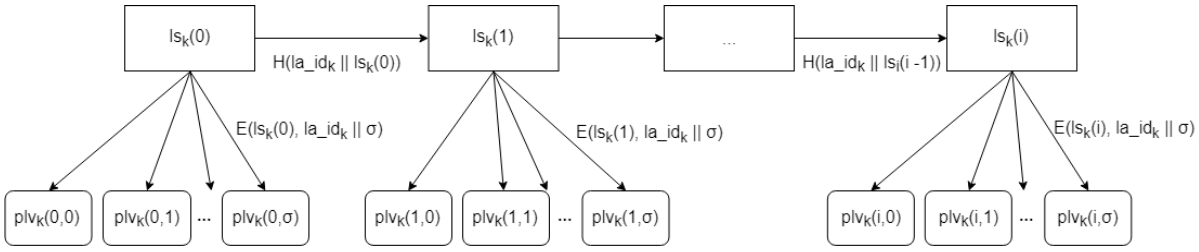
O encadeamento de certificados é feito com a inserção de valores de encadeamento (*linkage values*) em cada um dos certificados. Isso permite a revogação em lote de qualquer conjunto de pseudônimos cujo período de validade não tenha expirado.

No estudo original, são utilizadas duas entidades de encadeamento LA e a generalização entre elas é feita através da nomenclatura LA_k, com $k = 1$ ou $k = 2$. Cada entidade LA_k é identificada através de uma cadeia única de 32 bits, denominada la_id_k. Para definir o período de validade dos certificados, o tempo na rede é dividido em intervalos (e.g. uma semana) e cada intervalo é identificado por um valor i, com $i > 0$. Dessa forma, $i = 1$ corresponde à primeira semana do veículo a partir da obtenção de pseudônimos, $i = 2$ corresponde à segunda semana, e assim por diante.

A pedido da RA, a LA_k obtém uma semente pseudoaleatória de 128 bits $ls_k(0)$, válida para o período de tempo $t = 0$. O valor da semente $ls_k(t)$ para os períodos seguintes é calculado por $ls_k = H(la\_id_k \mid\mid ls_k(i-1))$. O estudo SCMS define a função $H(x)$ como os 16 bits mais significativos da função de hash SHA-256 aplicada sobre x. A função $a \mid\mid b$ é definida como o resultado da concatenação das cadeias de bits a e b.

A partir de cada semente $ls_k(t)$, são gerados σ valores de pré-encadeamento (*pre-linkage values*) $plv_k(i,\ j)$, com σ correspondendo o número de certificados válidos simultaneamente em um período de tempo t. O valor $j > 0$ corresponde ao número de identificação do certificado dentro de seu lote de tamanho σ. Os valores de pré-encadeamento são calculados através de $plv_k(i, j) = E(ls_k(i),\ la\_id_k \mid\mid j)$. A operação $E(k,\ m)$ é definida como os 8 bits mais significativos da saída do algoritmo AES ao cifrar a mensagem m com a chave k. A figura 6 exemplifica a obtenção dos valores de pré-encadeamento.

Figura 6: Obtenção dos valores de pré-encadeamento pelas entidade LA_k



Fonte: Autoria própria

Os valores de pré-encadeamento são cifrados individualmente com a chave pública da PCA e enviados para a RA. Assim, os $plv(t)$ são incluídos nas requisições à PCA sem que haja a descoberta de seus valores por parte da RA. A PCA recebe os $plv(t)$ de ambas as LAs e gera os valores de encadeamento (*linkage values*) através de $lv(t) = plv_1(t) \oplus plv_2(t)$, no qual $\oplus$ representa a operação de "ou"exclusivo, conhecida como XOR. Por fim, a PCA inclui $lv(t)$ no pseudônimo antes de assiná-lo, permitindo que seu valor seja visualizado por qualquer um. Os $plv(t)$, no entanto, devem ser mantidos em segredo e podem apenas ser revelados para a MA se a revogação do certificado for solicitada.

2.2.2.3 Propostas de melhoria no estudo SCMS

Desde sua publicação em 2013, o estudo SCMS têm ganhado propostas de melhorias e modificações, de forma a fortalecer sua proposta. O estudo BCAM (KUMAR; PETIT; WHYTE, 2017), por exemplo, mencionado na seção 2.2.1.4, tem sua estrutura baseada na solução SCMS. Esta decisão tomada pelos autores do artigo permite explorar novos mecanismos de segurança e, ao mesmo tempo, desfrutar das vantagens relacionadas à privacidade e escalabilidade proporcionadas pelo SCMS.

O estudo apresentado por Jr et al. (2018) levanta questionamentos sobre as deficiências do processo de encadeamento de certificados no sistema SCMS e propõe alterações para o modelo existente. As modificações permitem a revogação temporária de certificados, ao contrário da revogação vitalícia oferecida originalmente. Além disso, o nível de privacidade fornecido ao usuário é maior do que comparado ao estudo original.

Em 2018, o USDOT publicou um novo artigo com modificações no sistema SCMS original. Brecht et al. (2018) especificam novos detalhes relativos à revogação de certificados, além de aprimorar outros processos não descritos neste documento.

Assim, a proposta SCMS tem recebido diversas contribuições e alterações, inclusive pelos próprios autores do artigo. A partir disso, conclui-se que, apesar de o sistema original atender aos requisitos relacionados à emissão de certificados em comunicação veicular, ainda há espaço para melhorias.

3 ESPECIFICAÇÃO DO SISTEMA

A validação dos mecanismos de privacidade do modelo SCMS é feita através do desenvolvimento do protótipo de um sistema de emissão de certificados para comunicação veicular. O protótipo permite simular a solicitação e fornecimento de pseudônimos sob o ponto de vista de quatro perfis diferentes: veículo, PCA, RA e LA. Cada perfil é representado por uma aplicação separada, que possui suas próprias configurações de comunicação e banco de dados. Isso garante o isolamento de informações entre as entidades representadas.

3.1 Modos de operação

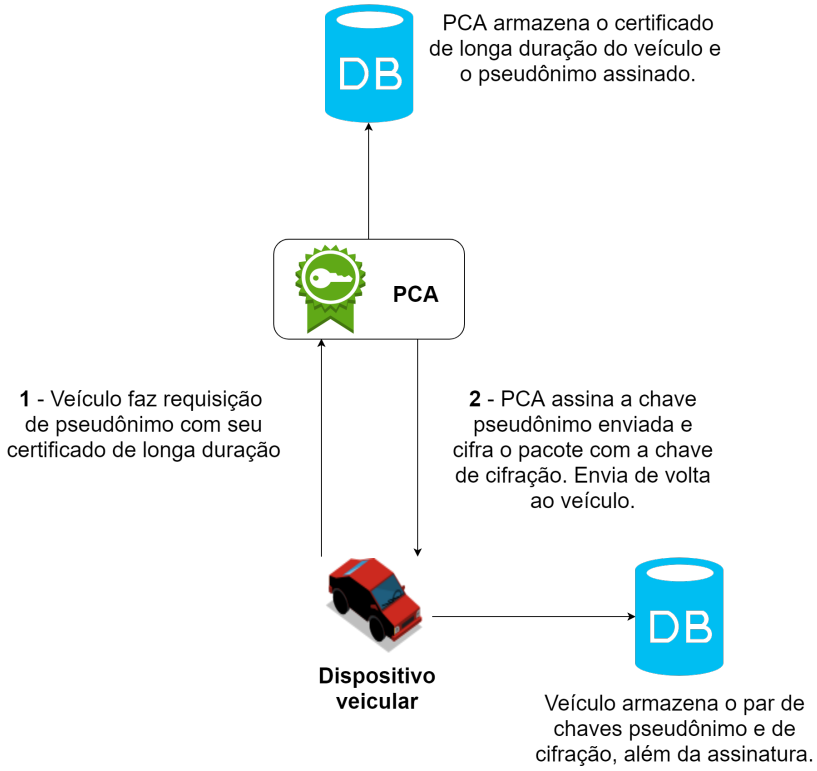
Com a definição dos perfis, a comparação entre os modelos de emissão de certificado é feita através de três modos de operação, que também são configurados na inicialização do sistema. O modo de operação da subseção 3.1.1 possui apenas os perfis veículo e PCA, enquanto que os modos das subseções 3.1.2 e 3.1.3 possuem os perfis veículo, PCA e RA disponíveis para seleção. O perfil LA somente está disponível no terceiro modo de operação.

3.1.1 Primeiro modo de operação - Sem privacidade

O primeiro modo de operação segue as características de uma PKI tradicional, como descrita na seção 2.1.4. Nesse contexto, o veículo solicita diretamente à entidade emissora um único pseudônimo, sem que haja qualquer proteção à identidade do usuário. O veículo gera dois pares de chaves assimétricas, um para o pseudônimo e outro para cifrá-lo. Ambas as chaves públicas são entregadas cifradas para a PCA. A PCA assina a chave pseudônimo e utiliza a outra chave para cifrar o pacote final. Neste modelo, a PCA tem acesso à identidade dos veículos e aos seus respectivos pseudônimos, sendo possível o rastreio dos usuários à medida em que ocorre a comunicação veicular. Além disso, a obtenção de múltiplos pseudônimos implica em múltiplas solicitações à PCA, o que aumenta o

processamento realizado pelo veículo. A figura 7 mostra o comportamento do sistema no primeiro modo de operação.

Figura 7: Processo de obtenção de certificados no primeiro modo de operação



Fonte: Adaptado de (WHYTE et al., 2013)

3.1.2 Segundo modo de operação - SCMS sem eficiência

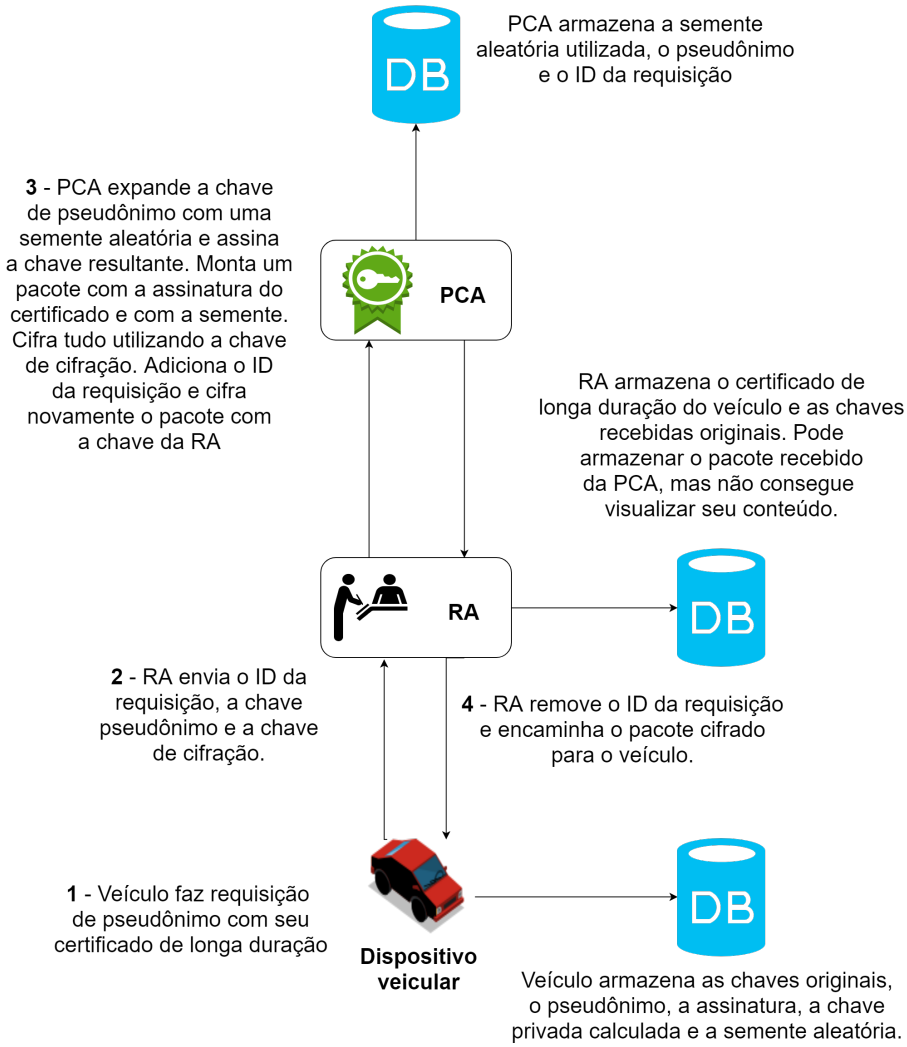
O segundo modo de operação aplica o modelo SCMS parcialmente. É adicionada a entidade do sistema SCMS correspondente à RA, que permite ocultar a identidade do veículo e garantir sua privacidade. O veículo solicita um único pseudônimo através da entrega de duas chaves públicas cifradas para a RA. O veículo também fornece uma chave pública correspondente à sua identidade de longo prazo. A RA recebe esses parâmetros e, após a decifração, cria uma solicitação à PCA. A identidade real do veículo é removida e é utilizado um identificador de solicitação, que corresponde ao Hash entre a identidade do veículo e à chave pública destinada ao pseudônimo. A cada solicitação, a PCA gera um novo pseudônimo a partir da expansão de uma das chaves públicas recebidas.

O pacote de retorno contém a assinatura do pseudônimo final e o valor aleatório utilizado na expansão. Esses dados são cifrados com a segunda chave enviada pelo veículo. A mensagem final contém os dados cifrados e o identificador de solicitação, e esta men-

sagem deve ser cifrada novamente, desta vez com a chave pública da RA. Ao receber o pacote, a RA não consegue ler o conteúdo do certificado, mas consegue identificar a qual veículo ele pertence, e o pacote é reencaminhado. Por fim, o veículo recebe a assinatura do pseudônimo, juntamente com os parâmetros de expansão, o que permite a obtenção das chaves privadas para a utilização do pseudônimo.

No segundo modo de operação, os certificados são solicitados individualmente, e não em lote. Dessa forma, não há encadeamento dos pseudônimos e não é possível revogá-los em lote. A figura 8 mostra o comportamento do sistema no segundo modo de operação.

Figura 8: Processo de obtenção de certificados no segundo modo de operação



Fonte: Adaptado de (WHYTE et al., 2013)

3.1.3 Terceiro modo de operação - SCMS completo

O terceiro modo de operação atende a todos os requisitos do modo anterior (seção 3.1.2), mas inclui os procedimentos de expansão de chaves borboleta e de encadeamento de chaves, de acordo com a descrição das seções 2.2.2.1 e 2.2.2.2. A expansão de chaves permite ao veículo obter todos os pseudônimos necessários com apenas uma única solicitação.

As principais alterações neste modo de operação referem-se ao comportamento das entidades RA e PCA e à adição de duas entidades LA. A ra recebe os dados cifrados e aplica a expansão de chaves borboleta. A RA também passa a solicitar a cada uma das LAs valores de pré-encadeamento, de forma que haja um valor para cada expansão obtida. Ao receber tais valores, a RA cria solicitações individuais à PCA correspondentes a cada uma das expansões e anexa os correspondentes valores cifrados de pré-encadeamento.

A PCA submete os valores de pré-encadeamento à operação "OU"exclusivo, o que dá origem ao valor de encadeamento (*linkage value*). Este valor é adicionado ao pseudônimo e pode ser visualizado abertamente. As alterações propostas introduzem o encadeamento de chaves, que permite a revogação eficiente dos certificados pertencentes a usuários maliciosos. A figura 9 mostra o comportamento do sistema no terceiro modo de operação.

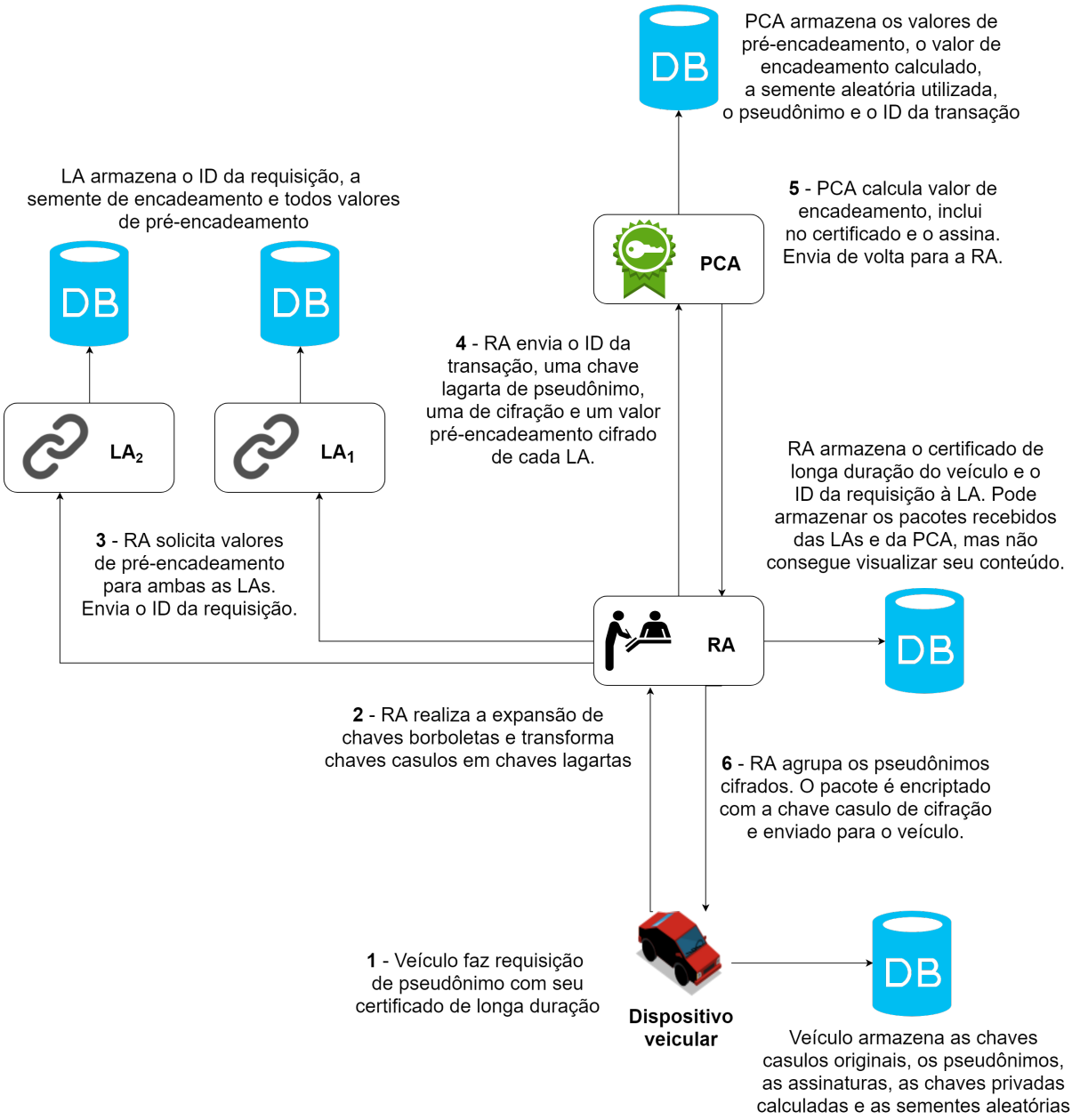
3.2 Definição do escopo e simplificações do sistema

Por ser um trabalho individual com tempo limitado de desenvolvimento, é importante definir o escopo do projeto e quais simplificações foram adotadas. O escopo definido conta com uma única versão de cada entidade do sistema, com exceção da entidade LA. Em um sistema real, várias cópias de cada entidade existiriam, para atender à demanda de regiões maiores. O embaralhamento de requisições feito pela RA também foi removido do escopo do projeto, uma vez que aumenta a complexidade da comunicação entre sistemas. Apesar disso, é uma característica que deve estar presente em uma implementação real do sistema.

Algumas modificações foram feitas no sistema SCMS para facilitar seu desenvolvimento. Dentre elas, podemos citar:

- A entidade RA não valida o certificado de identidade do veículo;
- Não é feita a verificação se o veículo solicitou pseudônimos recentemente, tanto para

Figura 9: Processo de obtenção de certificados no terceiro modo de operação



Fonte: Adaptado de (WHYTE et al., 2013)

a mesma entidade RA quanto para outra de suas cópias (em um cenário no qual existem múltiplas RAs);

- Não existe uma lista de certificados e veículos banidos da rede;
- Os certificados das entidades são estáticos e pré-configurados na aplicação, ou seja, não há uma infraestrutura de chaves públicas para consultar a chave de cada uma das entidades. Apenas confia-se que a chave pública instalada é aquela.

Neste projeto, está fora do escopo a proposta de modificações originais no sistema SCMS. Esta tarefa é reservada para o possível período de mestrado.

3.3 Métodos empregados

A primeira etapa deste trabalho foi o levantamento dos aspectos conceituais e o estado da arte relativo à comunicação veicular. Foram levantados diversos artigos sobre comunicação entre veículos (*Vehicle-to-Vehicle* - V2V) e entre veículos e infraestrutura (*Vehicle-to-Infrastructure* - V2I), ambas as classificações comumente agrupadas como comunicação entre veículos e "tudo"(*Vehicle-to-Everything* - V2X). A revisão de literatura é parte fundamental deste trabalho para compreender o cenário atual de comunicação V2X, de forma a avaliar se a solução SCMS escolhida é, de fato, a mais promissora. O estudo de tais soluções é parte integral do projeto, uma vez que novas propostas podem ser publicadas ao longo do ano, e, portanto, é natural que esta etapa ocupe boa parte do projeto de formatura.

A partir do estudo da solução SCMS, foi realizada a descrição dos requisitos do sistema, sejam esses funcionais ou não-funcionais. A descrição do funcionamento do sistema deve ser elaborada tendo em vista o comportamento do veículo e das outras entidades do modelo. É necessário definir os principais requisitos e funções de cada uma das aplicações e como é feita a comunicação entre elas, através da especificação de protocolos e do formato das mensagens.

Concomitante ao levantamento de requisitos, foi feito o levantamento das ferramentas necessárias ao projeto. Para o desenvolvimento das aplicações, é necessário escolher linguagens de programação com bibliotecas criptográficas flexíveis, que permitam a execução de funções criptográficas de baixo nível. Há necessidade de gerar chaves criptográficas seguindo parâmetros muito específicos; no contexto de curvas elípticas, é necessário ter

controle sobre os pontos da curva utilizados, o valor do escalar pelo qual o ponto é multiplicado, entre outros fatores. Assim, um levantamento detalhado sobre as linguagens e suas bibliotecas foi realizado, seguido por decisões relativas a banco de dados e comunicação entre aplicações.

Em sequência, tem-se início ao desenvolvimento do sistema em si, utilizando como base a documentação. Foram utilizadas ferramentas de gerenciamento de código, como Git, para o acompanhamento de seu progresso. É importante ressaltar que a documentação progride juntamente com a etapa de desenvolvimento, tanto para a elaboração deste documento quanto na descrição das funções do código.

A próxima etapa do projeto consiste na realização de testes e análises de funcionamento do sistema. É necessário um conjunto minucioso de testes para garantir que os mecanismos criptográficos estão sendo utilizados adequadamente.

Na última etapa, são levantadas métricas de desempenho e demonstrações relativas à privacidade do modelo SCMS. A partir desse ponto, são retiradas as conclusões pertinentes ao projeto e levantadas propostas de continuidade.

As atividades definidas para este projeto e seu período de execução são exibidos na tabela 1.

Tabela 1: Cronograma de atividades para o projeto

Atividade	**Jan**	**Fev**	**Mar**	**Abr**	**Mai**	**Jun**	**Jul**	**Ago**	**Set**	**Out**	**Nov**	**Dez**
Leitura de artigos sobre comunicação veicular	X	X	X	X	X	X	X					
Estudo da solução SCMS		X	X	X	X	X						
Levantamento dos requisitos do sistema				X	X	X	X					
Levantamento de ferramentas e bibliotecas utilizadas no projeto					X	X	X					
Especificação da documentação do sistema				X	X	X	X	X				
Desenvolvimento do sistema						X	X	X	X	X	X	
Testes do sistema										X	X	X
Análise do sistema e retirada de métricas pertinentes										X	X	X
Definição das conclusões obtidas com o projeto											X	X
Registro na monografia da evolução da aplicação						X	X	X	X	X	X	X
Entrega do projeto												X

Fonte: Autoria própria

3.4 Requisitos funcionais

Os requisitos funcionais do sistema são relativos às atividades que o sistema deve desempenhar. O funcionamento do sistema depende de cinco aplicações em execução: uma simula o comportamento do veículo, uma simula o comportamento da PCA, uma simula o comportamento da RA e duas simulam o comportamento da LA.

Os principais requisitos referem-se à possibilidade de enviar mensagens pela rede, cifrar e decifrar informações e realizar expansões criptográficas. Mais detalhes são apresentados nas seções 3.4.1, 3.4.2, 3.4.3 e 3.4.4.

3.4.1 Veículo

As principais operações realizadas pelo veículo veículo correspondem à geração de chaves criptográficas, além de sua manipulação através de operações relacionadas a curvas elípticas. Também deve ser capaz de cifrar e decifrar mensagens, utilizando um algoritmo de curvas elípticas simétrico, e deve ser capaz de comunicar-se com as entidades PCA e RA. O armazenamento das solicitações também é um dos requisitos para esta entidade.

3.4.2 Autoridade registradora (RA)

As principais operações realizadas pela RA correspondem à manipulação de pontos e inteiros pertencentes à uma curva elíptica, de forma a realizar a expansão criptográfica. Deve ser capaz de cifrar e decifrar mensagens, utilizando o algoritmo AES e um algoritmo de curvas elípticas simétrico, e deve ser capaz de comunicar-se com o veículo, com a PCA e com as LAs. Também deve armazenar as informações recebidas de cada uma das entidades.

3.4.3 Autoridade certificadora de pseudônimos (PCA)

Assim como a RA, as principais operações realizadas pela PCA correspondem à manipulação de pontos e inteiros pertencentes à uma curva elíptica, de forma a gerar os pseudônimos finais. Deve ser capaz de cifrar e decifrar mensagens com um algoritmo de curvas elípticas simétrico, e deve ser capaz de comunicar-se com o veículo e com a RA, de acordo com o modo de operação. Deve ser capaz de assinar certificados utilizando sua chave pública, o que implica também a obtenção do Hash do certificado. Deve realizar o armazenamento em seu banco de dados das informações recebidas.

3.4.4 Autoridade encadeadora (LA)

Os requisitos funcionais associados à autoridade encadeadora referem-se à possibilidade de gerar números pseudoaleatórios, obter valores de hash e cifrar valores utilizando cifras simétricas e assimétricas. A comunicação também é fator importante para o funcionamento desta entidade, uma vez que a LA recebe solicitações da RA para gerar valores de pré-encadeamento. Uma vez obtidos tais valores, eles são enviados cifrados de volta à RA.

3.5 Requisitos não funcionais

Os requisitos não funcionais estão relacionados ao uso da aplicação e a características definidas para seu funcionamento. Os principais requisitos não funcionais definidos para esta aplicação são: privacidade, confidencialidade, irretratabilidade, autenticidade e escalabilidade.

3.5.1 Privacidade

Um dos principais requisitos não funcionais é a privacidade do usuário. O principal objetivo do sistema é garantir a anonimidade condicional do dispositivo, de forma que ele não possa ser rastreado. As informações pessoais devem ser mantidas em sigilo para que localizações-chave do usuário não sejam descobertas, como, por exemplo, o endereço de sua casa ou de seu trabalho. Deve-se observar que, ao garantir privacidade, o sistema está protegido contra ataques internos aos dados dos usuários, ou seja, ataques feitos por entidades do sistema que sejam maliciosas ou "honestas mas curiosas"(KISSNER; SONG, 2005).

3.5.2 Confidencialidade

O requisito de confidencialidade deve garantir que as informações transmitidas na rede são reveladas apenas aos usuários a que se destinam. Ao garantir confidencialidade, deseja-se proteger o sistema contra ataques externos, no qual o atacante não pertence à rede, mas meramente a utiliza para obter as informações desejadas. Este requisito é atendido através da cifração das mensagens transmitidas.

3.5.3 Irretratabilidade

A irretratabilidade é um fator importante em redes veiculares devido à necessidade de punir usuários mal-intencionados. O sistema deve garantir a descoberta da identidade de todos os usuários da rede e a associação de suas identidades às suas ações. Essa descoberta deve ocorrer apenas quando há violação de conduta da rede e, acima de tudo, não deve ser realizada por apenas uma entidade. A descoberta deve ser possível apenas com a colaboração de duas entidades da PKI e essa operação deve ser supervisionada por uma terceira entidade. O intuito desse procedimento é justamente proteger o requisito de privacidade, descrito na seção 3.5.1.

3.5.4 Autenticidade

A autenticidade garante que o remetente de uma mensagem seja identificado pelo seu destinatário. No cenário de redes veiculares, o usuário deve provar que pertence à rede e que tem permissão para participar dela antes de enviar qualquer mensagem básica de segurança. A autenticação pode ser feita através dos pseudônimos, omitindo, assim, a verdadeira identidade do veículo.

3.5.5 Escalabilidade

O sistema proposto busca simular o comportamento de uma rede veicular. Dessa forma, devem ser discutidos os requisitos envolvidos em uma aplicação real desta rede. O modelo de comunicação que for adotado como padrão para redes veiculares deve garantir o requisito de escalabilidade, uma vez que futuramente será utilizado em todos os veículos e dispositivos de infraestrutura, como semáforos, radares, entre outros equipamentos.

Como explicado na seção 2.2.2, o sistema SCMS em capacidade máxima deverá emitir aproximadamente 300 bilhões de certificados por ano para um total de 300 milhões de veículos. Apesar de o requisito de escalabilidade não ser necessário para o desenvolvimento deste protótipo simplificado, é importante ter em mente que alterações feitas no sistema SCMS devem aumentar ou manter a escalabilidade e eficiência atuais.

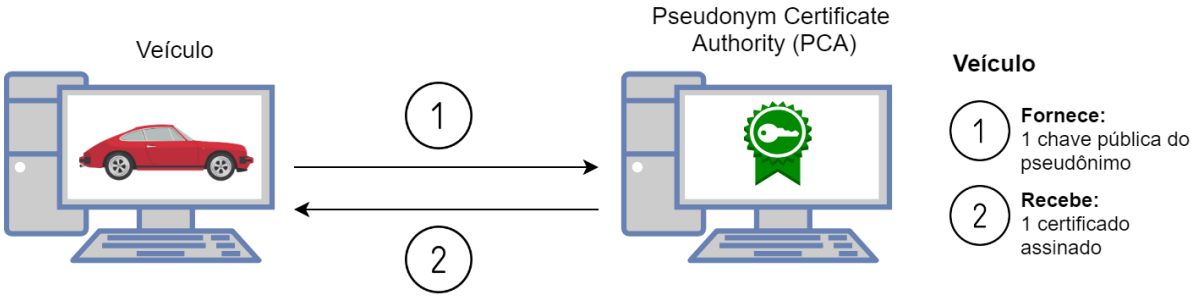
3.6 Arquitetura do sistema

A partir da definição comportamental da aplicação, é possível definir a arquitetura do sistema para o funcionamento das aplicações. Cada entidade simulada do sistema

SCMS deve ser executada individualmente e ocupar uma porta diferente na comunicação (mais detalhes na seção 4.2.1). Os componentes do sistema devem ser executados em computadores desktop ou em notebooks. Uma vez que o trabalho aqui presente é apenas um protótipo para a validação de hipóteses, as aplicações também podem ser executadas em um único dispositivo, desde que haja separação entre os bancos de dados e as portas de execução.

O primeiro modo de operação necessita de dois dispositivos para a simular o comportamento do veículo e da PCA. A figura 10 representa a arquitetura de tal sistema e identifica a ordem da interação de seus participantes. O veículo solicita a assinatura de uma única chave pública e a PCA retorna um certificado assinado.

Figura 10: Arquitetura da aplicação no primeiro modo de operação - sem privacidade

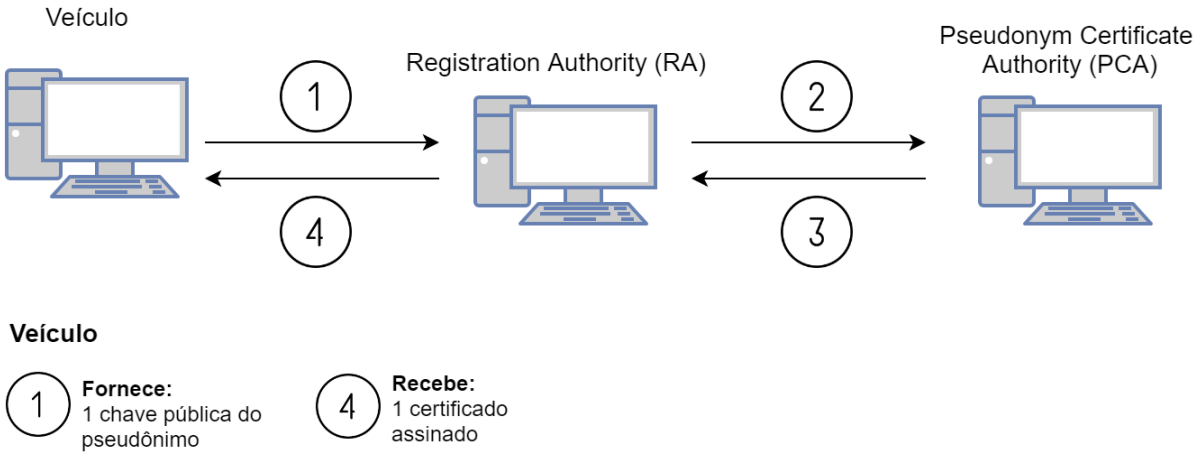


Fonte: Autoria própria

O segundo modo de operação necessita de três dispositivos, que representam o comportamento do veículo, da RA e da PCA. A arquitetura do sistema está representada na figura 11, juntamente com a ordem das operações realizadas. Da mesma forma que o primeiro modo de operação, o veículo fornece apenas uma chave pública do pseudônimo e recebe apenas um certificado.

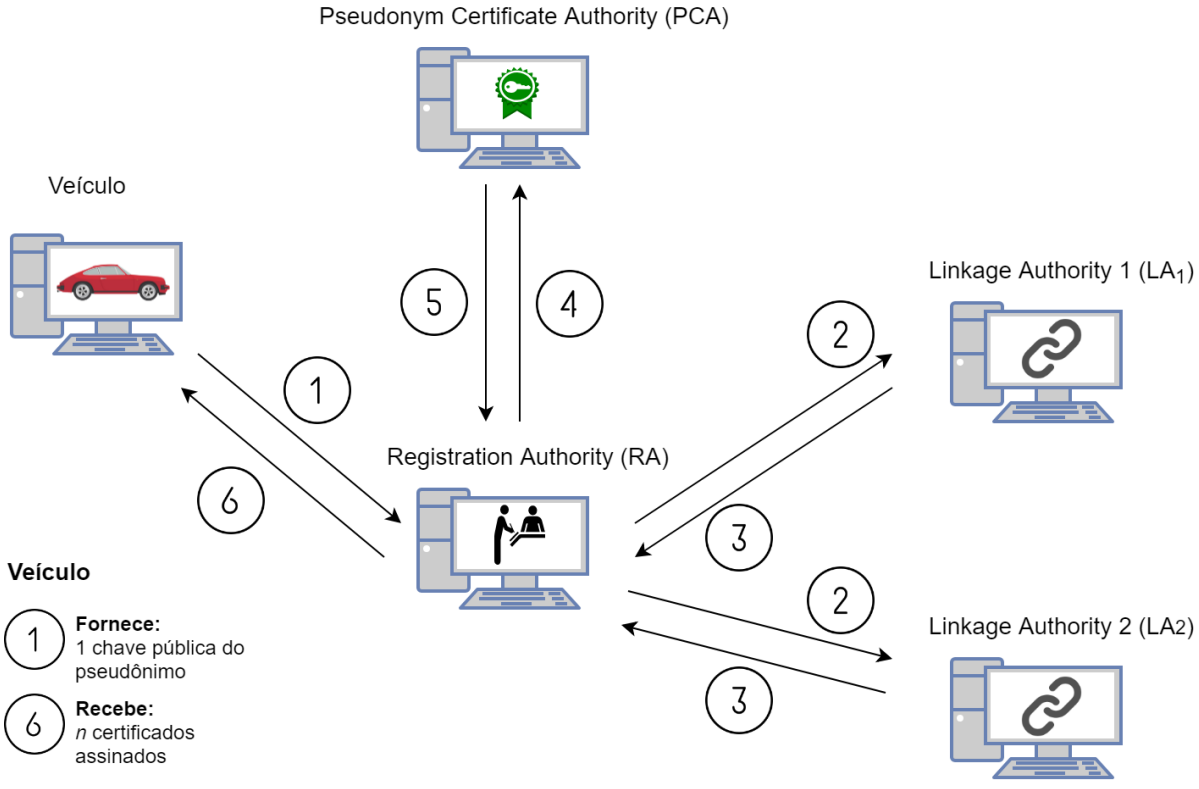
O terceiro modo de operação necessita de cinco dispositivos, que representam o comportamento do veículo, da RA, da PCA e de duas LAs. A figura 12 exibe a arquitetura do sistema e a ordem de interação entre entidades. A principal diferença neste modo de operação refere-se aos dados recebidos pelo usuário final. O veículo realiza uma única solicitação, fornecendo uma chave pública do pseudônimo, e em troca recebe n certificados assinados. O custo de processamento para o veículo é, portanto, menor do que nos outros casos.

Figura 11: Arquitetura da aplicação no segundo modo de operação - privacidade sem eficiência



Fonte: Autoria própria

Figura 12: Arquitetura da aplicação no terceiro modo de operação - privacidade com eficiência



Fonte: Autoria própria

4 TECNOLOGIAS UTILIZADAS

4.1 Linguagem de programação

A escolha da linguagem de programação para o desenvolvimento de uma aplicação deve levar em consideração todas as necessidades e requisitos do projeto, assim como a disponibilidade de bibliotecas e documentações de apoio. Para este projeto, a escolha de uma boa biblioteca criptográfica foi o ponto de partida para as decisões relativas à linguagem de programação. A definição de uma "boa biblioteca criptográfica"envolve o acesso a funções primitivas de criptografia, que permitam o controle preciso de seus parâmetros. Bibliotecas tradicionais apenas permitem a geração de pares de chaves criptográficas, sem que ocorra manipulação de seus parâmetros. No entanto, a expansão de chaves borboleta, vista na seção 2.2.2.1, realiza operações de "baixo nível"criptográfico no contexto de curvas elípticas, como a multiplicação de um ponto da curva por um escalar. A biblioteca escolhida, portanto, deve fornecer controle granular sobre todos os seus parâmetros.

A biblioteca que melhor atende a todos os requisitos é a RELIC Toolkit (ARANHA; GOUVêA, 2009), uma biblioteca com ênfase em eficiência e flexibilidade. RELIC foi desenvolvida em C e possui implementações de curvas elípticas, o que a torna adequada para este projeto.

O uso da linguagem C, no entanto, dificulta a implementação do projeto ao torná-la mais verbosa e mais propensa a erros, uma vez que são utilizadas construções como ponteiros. Ao considerarmos a utilização de uma interface *web*, esta dificuldade aumenta ainda mais. Por causa disso, buscou-se alternativas para a linguagem de programação que permitisse a utilização da biblioteca RELIC.

Após tais considerações, a linguagem de programação escolhida para a aplicação é Python (PYTHON, 2001), em sua versão 3.6.5. Experiências anteriores com a linguagem auxiliaram no processo de escolha, mas a possibilidade de importar bibliotecas escritas em C e a chamada de suas funções diretamente no código Python foi o fator diferencial. A aplicação possui interface *web*, desenvolvida através do *framework* Django (DJANGO,

2005). A escolha também foi baseada em experiências anteriores, além de ser a principal ferramenta para desenvolver aplicações *web* na linguagem Python.

4.2 Outras tecnologias

Esta seção descreve outras tecnologias, bibliotecas ou ferramentas utilizadas no desenvolvimento do projeto. É dado destaque ao processo de comunicação e armazenamento de dados.

4.2.1 Comunicação

A comunicação entre as aplicações é feita através do uso de sockets, que são uma interface de software entre as camadas de aplicação e de transporte de um computador. A comunicação é feita através do endereçamento do hospedeiro e do identificador de processo, que correspondem, respectivamente, ao endereço de IP e ao número da porta em que o processo do destinatário está sendo executado. Foram utilizadas bibliotecas de WebSocket, que permitem a configuração da comunicação de forma mais rápida.

Para a entidade que atuar como servidor, como é o caso da PCA, por exemplo, foi utilizada a biblioteca Channels (FOUNDATION, 2018), que corresponde a uma extensão do framework Django. Para a entidade que atuar como cliente, como é o caso do veículo, por exemplo, foi utilizada a biblioteca Websocket-client (WEBSOCKET-CLIENT, 2018).

4.2.2 Armazenamento

O armazenamento de informações da aplicação é feito através de um banco de dados de objetos relacionais com o gerenciador PostgreSQL (POSTGRESQL, 1996). O sistema gerenciador PostgreSQL é extremamente robusto e confiável, além de ser flexível e rico em recursos. Suas definições atendem aos requisitos levantados para o projeto.

5 PROJETO, IMPLEMENTAÇÃO E TESTES

Este capítulo descreve as etapas de desenvolvimento do projeto e as decisões tomadas durante seu percurso. São discutidos alguns detalhes de implementação, de forma que o leitor possa compreender o funcionamento do código. Ao final, são apresentados os testes realizados no sistema e os resultados obtidos.

5.1 Etapas de projeto

A partir da especificação do projeto, é possível dividi-lo em quatro etapas principais. A primeira etapa consiste no encapsulamento da biblioteca RELIC Toolkit para Python e sua consequente integração. As próximas etapas consistem no desenvolvimento de cada um dos três módulos de operação. O código para os três modos de operação encontra-se disponível no link ⟨https://bitbucket.org/scmssimulatorteam/⟩. As próximas seções descrevem a evolução de cada uma das etapas e suas consequentes decisões de projeto.

5.1.1 Integração com a biblioteca RELIC Toolkit

A integração com a biblioteca RELIC Toolkit foi feita em projeto python à parte, para inclusão no código principal através da importação de módulos. O projeto, denominado "pyrelic", é formado pelo código Python encarregado do encapsulamento e também pelo código C modificado da biblioteca RELIC. O código do módulo pyrelic está disponível em ⟨https://bitbucket.org/giulianabb/pyrelic_toolkit/⟩. As modificações e decisões tomadas em relação à biblioteca original são discutidas ao longo desta seção.

Para o encapsulamento da biblioteca RELIC Toolkit, foi utilizada a biblioteca ctypes (FOUNDATION, 2018), que permite a chamada de funções de uma biblioteca C em Python, seja esta estática ou compartilhada. Para facilitar a configuração, compilação e chamada da biblioteca RELIC, foi utilizado o sistema operacional Ubuntu (UBUNTU, 2018) em sua versão 18.04 LTS.

A compilação da RELIC é feita através da ferramenta CMake (KITWARE, 2018). Esta ferramenta permite configurar rapidamente os parâmetros da biblioteca através de *presets*, ou seja, um conjunto de configurações pré-definidas. As configurações utilizadas para este projeto são especificadas na tabela 2.

Tabela 2: Configurações utilizadas pelo CMake na compilação da biblioteca RELIC

Configuração	**Valor**
WORD	64
RAND	HASH
MD_METHD	SH256
SHLIB	ON
STBIN	OFF
TIMER	CYCLE
CHECK	off
VERBS	off
ARITH	x64
FP_PRIME	255
FP_METHD	INTEG;INTEG;INTEG;MONTY;LOWER;SLIDE
COMP	-O3 -funroll-loops -fomit-frame-pointer -finline-small-functions -march=native -mtune=native -fPIC
FP_PMERS	off
FP_QNRES	off
FPX_METHD	INTEG;INTEG;LAZYR
PP_METHD	LAZYR;OATEP

Fonte: Autoria própria

Após a compilação, é gerado o arquivo *librelic.so*, correspondente à biblioteca compartilhada. Este arquivo é chamado no código Python e, através dele, é possível acessar as funções do código C. É importante ressaltar que a biblioteca compartilhada apenas fornece acesso às funções do código original, sendo impossível, por exemplo, acessar macros. Isso se deve pelo próprio processamento das macros pelo compilador, que recebe definições como no código 1 e substitui todas as ocorrências de FP_PRIME por seu valor correspondente.

Código 1: Definição de macro em C

```
#define FP_PRIME 255
```

No caso da RELIC, muitas funções são definidas através de macros, inclusive funções que devem ser utilizadas para este projeto. Dessa forma, foi feito um levantamento das principais macros e seu encapsulamento em funções, dentro do próprio código C. Dessa forma, macros como no código 2 foram convertidas em funções, como descrito no código

3. Algumas definições numéricas, como no exemplo do código 1, foram copiadas para o código Python, considerando que serão utilizadas para chamar as funções da biblioteca.

Código 2: Macro para a função de multiplicação de Big Number

```
#define bn_mul(C, A, B)          bn_mul_comba(C, A, B)
```

Código 3: Conversão da macro bn_mul para a função bn_mul_macro.

```
void bn_mul_macro(bn_t C, const bn_t A, const bn_t B) {
        bn_mul_comba(C, A, B);
}
```

Após as alterações na biblioteca, foi feito o encapsulamento das funções em Python, presente no arquivo */pyrelic/relic_wrapper.py*. A definição de constantes foi feita no arquivo */pyrelic/relic_constants.py* e funções auxiliares foram definidas em */pyrelic/relic_utils.py*. O arquivo */pyrelic/output_grabber.py* contém a lógica para obter a saída impressa no terminal pela função *printf* da biblioteca em C. Tal medida é útil para armazenar valores de saída de funções de impressão, como *bn_print*, que imprime o valor de um número inteiro de precisão múltipla (conhecido na biblioteca como *big number*).

Em */pyrelic/relic_wrapper.py*, são definidas seis classes principais de encapsulamento:

- **RelicUtil**: Classe estática que contém funções de inicialização da biblioteca RELIC. A função *core_init* deve ser chamada antes da utilização da biblioteca.
- **Hash**: Classe estática que contém a função de hash para o algoritmo SHA256, que foi selecionado como padrão nas configurações.
- **BigNumber**: Classe que representa a estrutura C conhecida como *bn_t*, ou *Big Number*. Possui as principais funções de criação e manipulação individual de inteiros de precisão múltipla, como a geração de valores aleatórios.
- **EllipticCurveConfiguration**: Classe estática que contém funções de configuração para o uso de curvas elípticas. A função *param_set_any* deve ser chamada obrigatoriamente antes da utilização de curvas elípticas.
- **Point**: Classe que representa a estrutura C conhecida como *ec_t*, ou *Elliptic Curve Point*. Contém as principais funções para criação e manipulação individual de pontos, além de funções estáticas para soma e multiplicação de pontos. As funções que transformam a representação do ponto para um valor binário utilizam compressão de ponto, de forma que o *array* produzido possua comprimento menor.

- **CryptoProtocols**: Classe estática que contém funções de criptografia com curvas elípticas, como geração de chaves, assinatura e verificação com o algoritmo ECDSA, além de cifração e decifração com o algoritmo ECIES.

É importante ressaltar que apenas as funções necessárias foram encapsuladas. Alguns parâmetros foram configurados manualmente no código utilizando valores fixos, o que impediria o uso da RELIC em outras configurações. Estas limitações foram impostas, principalmente, pelo tempo disponível para seu desenvolvimento.

5.1.2 Desenvolvimento do primeiro modo de operação

Para o primeiro modo de operação, foram criados dois projetos correspondentes às entidades veículo e PCA. O sistema do veículo é executado localmente na porta 7000, enquanto que o sistema da PCA é executado na porta 8000. A definição das portas é relevante para a comunicação entre as entidades.

Os pacotes enviados pelo veículo consistem em uma estrutura Python conhecida como dicionário, que contém os seguintes campos:

- *recovery_point*: corresponde ao binário do ponto para decifrar o texto recebido;
- *cyphertext*: corresponde às informações cifradas com a chave da PCA.

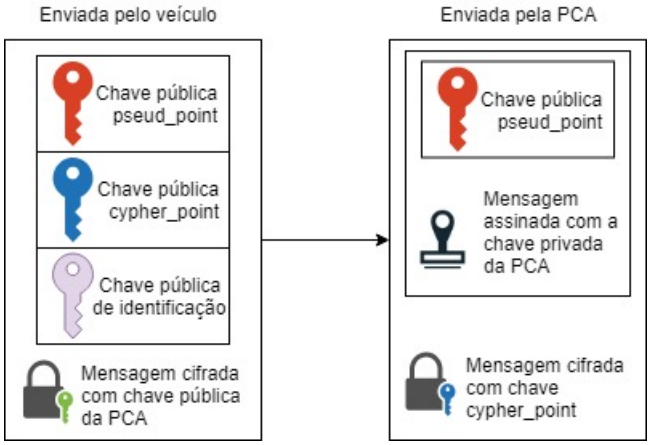
Os pacote são cifrados com a chave pública da PCA utilizando o algoritmo ECIES. Após a decifração, a PCA obtém o valor às claras de *cyphertext*, que corresponde a um dicionário com os seguintes campos:

- *pseud_point*: corresponde ao binário do ponto que deve ser assinado, ou seja, o valor da chave pública que dará origem ao pseudônimo;
- *cypher_point*: corresponde ao binário do ponto que deve cifrar o pacote de retorno.
- *vehicle_id*: corresponde ao binário do ponto representando a identidade do veículo. Cada veículo deve utilizar apenas um certificado de identificação, que corresponde ao seu certificado de longo prazo.

Através do algoritmo SHA-256, a PCA obtém o hash do pseudônimo e o assina utilizando o algoritmo ECDSA. A mensagem é cifrada para o veículo com o algoritmo ECIES,

utilizando a segunda chave enviada pelo veículo. Assim, o pacote de retorno possui os mesmos campos do pacote recebido, ou seja, *recovery_point* e *cyphertext*. Desta vez, *cyphertext* é um dicionário com os campos *sign_1* e *sign_2*, duas estruturas conhecidas como BigNumbers que correspondem à assinatura. É importante ressaltar que a verificação da assinatura requer o valor ambas as partes. A figura 13 mostra o formato das mensagens enviadas pelas entidades.

Figura 13: Formato das mensagens enviadas no primeiro modo de operação



Fonte: Autoria própria

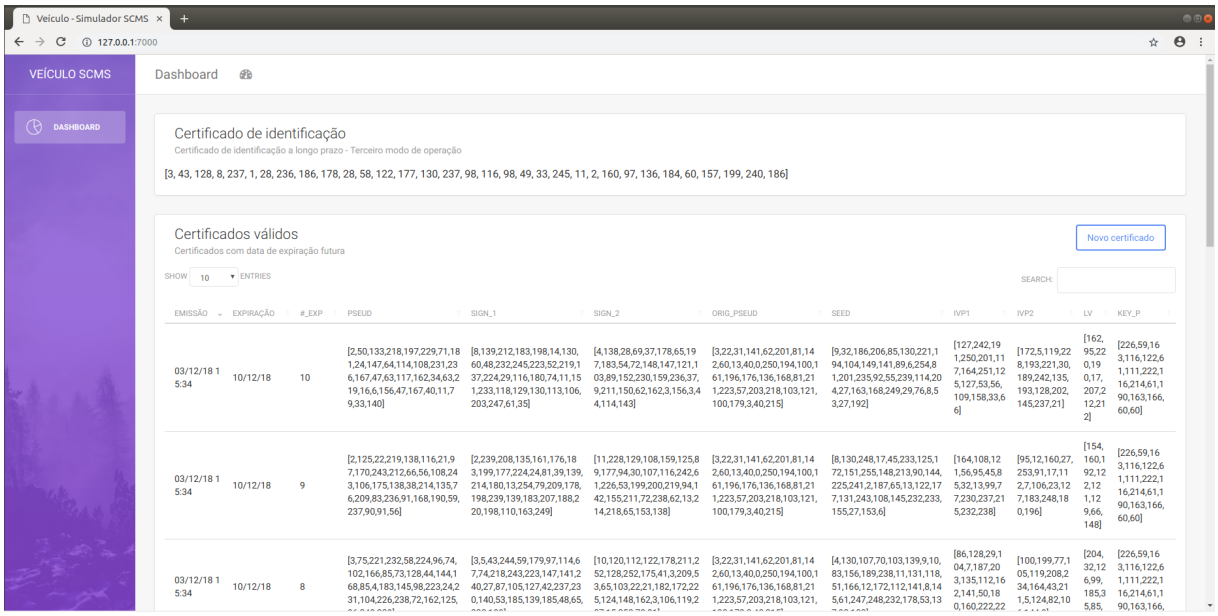
Os dados referentes à solicitação são armazenados pelo veículo e pela PCA. Tais dados são exibidos através de uma interface web em sua página principal, permitindo a comparação entre os valores obtidos por cada uma das entidades. As figuras 14 e 15 representam as interfaces web preliminares correspondentes ao veículo e à PCA, respectivamente.

5.1.3 Desenvolvimento do segundo modo de operação

Foi adicionado um novo projeto para a entidade RA, que é executada localmente na porta 9000. Os algoritmos utilizados para cifração das mensagens e assinatura de pseudônimos são os mesmos do primeiro modo de operação, ou seja, ECIES e ECDSA, respectivamente.

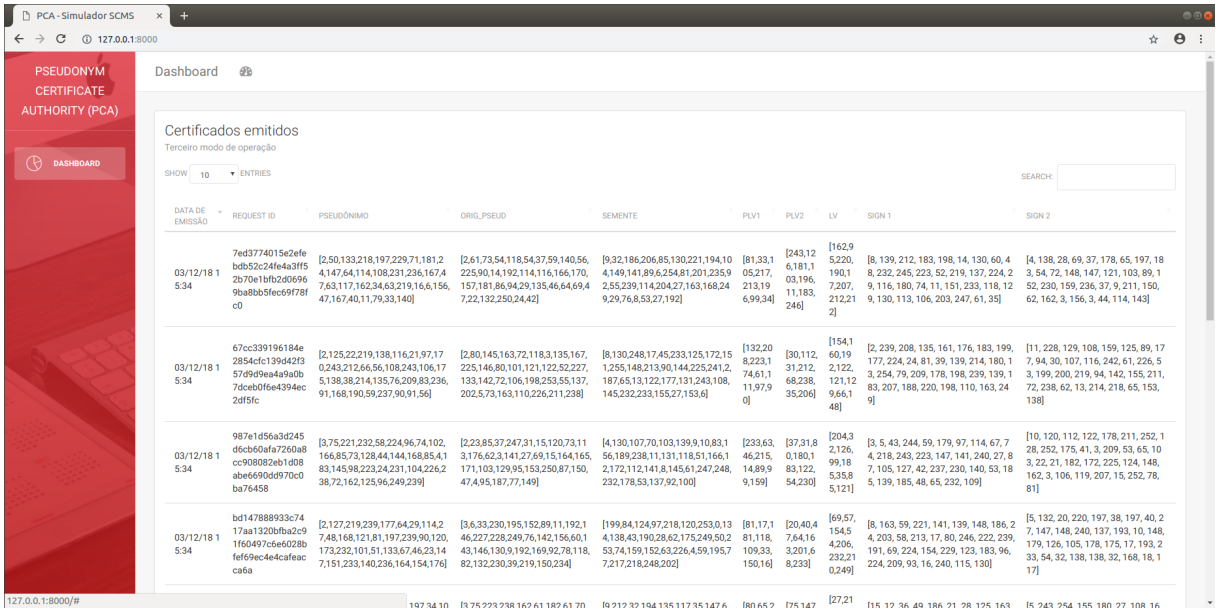
O veículo envia para a RA um pacote cifrado com os campos *cyphertext* e *recovery_point*. Após a decifração, a RA obtém um dicionário com os mesmos campos do primeiro modo de operação: *pseud_point*, *cypher_point* e *vehicle_id*. A RA remove o valor de *vehicle_id* e calcula o valor de *request_id*, correspondente ao Hash de *vehicle_id* concatenado a *pseud_point*. A mensagem é cifrada com a chave pública da PCA e enviada.

Figura 14: Interface web para o veículo



Fonte: Autoria própria

Figura 15: Interface web para a PCA

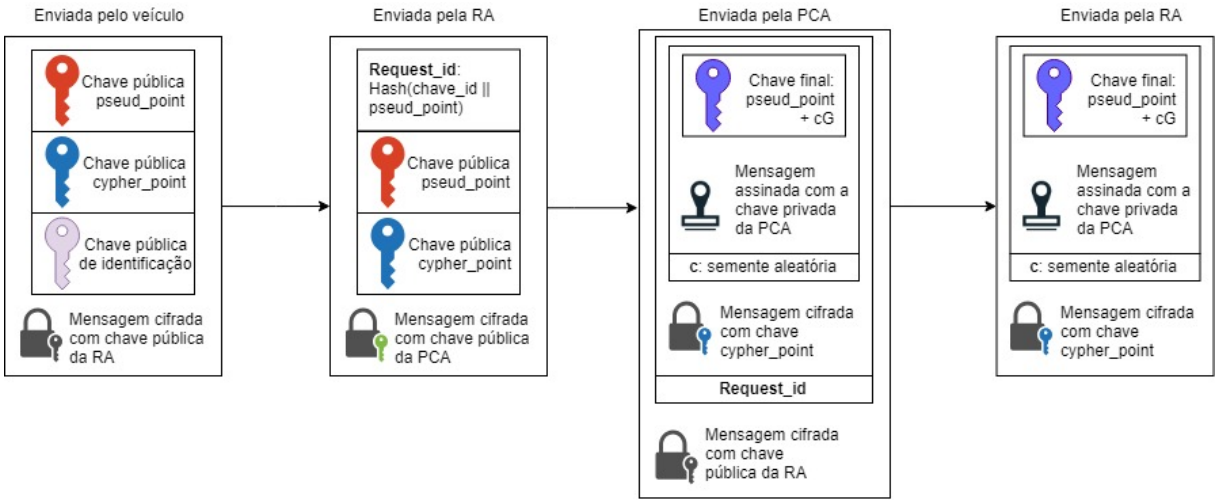


Fonte: Autoria própria

A PCA calcula o valor aleatório de c, obtendo o pseudônimo final do veículo. Este é assinado e cifrado com a chave *cypher_point*. Adiciona-se o valor de *request_id* à mensagem, que é cifrada com a chave pública da RA.

A RA decifra a mensagem, remove o *request_id* e retorna o valor restante ao veículo. A representação do formato das mensagens enviadas é apresentada na figura 16.

Figura 16: Formato das mensagens enviadas no segundo modo de operação



Fonte: Autoria própria

Após salvar os dados obtidos, estes são exibidos através de uma interface web. A interface correspondente à RA é exibida na figura 17.

5.1.4 Desenvolvimento do terceiro modo de operação

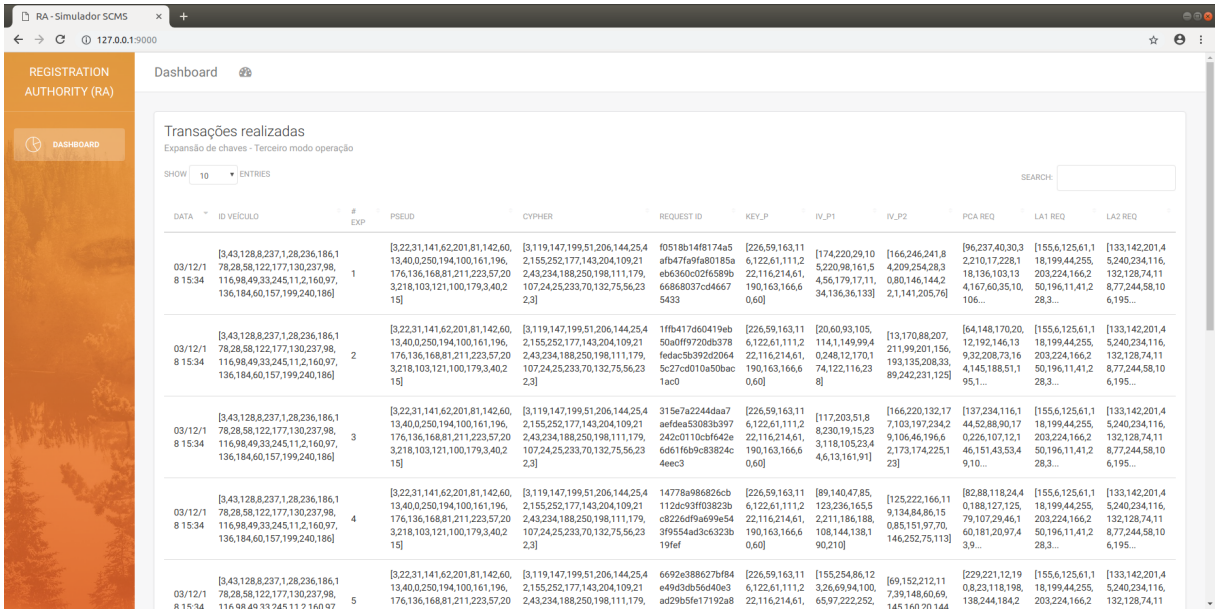
Um novo projeto foi adicionado para as entidades LA, de forma que possa ser executado duas vezes simultaneamente. O sistema da LA_1 é executado na porta 5000, enquanto que o sistema da LA_2 é executado na porta 5001. A assinatura de mensagens é feita com o algoritmo ECDSA e sua cifração é feita com o algoritmo ECIES. Também é utilizado o algoritmo AES para a expansão de chaves, sendo utilizado no modo de operação CBC.

O veículo gera dois pares de chaves criptográficas casulo, *pseud* e *cypher*, e duas funções de permutação pseudo-aleatória de inteiros, f_{kp} e f_{kc}. Na prática, podemos entender f_k como:

$$f_k(i) = AES(0_{128} \oplus i, key_x)||AES(1_{128} \oplus i, key_x) \tag{5.1}$$

no qual:

Figura 17: Interface web para a RA



Fonte: Autoria própria

- 0_{128} corresponde a um array de 0s com comprimento 128. O mesmo se aplica para 1_{128};
- $AES(m, k)$ corresponde à mensagem m cifrada com o algoritmo AES utilizando a chave k;

Assim, os valores que devem ser fornecidos pelo veículo correspondem às chaves de 128 bits que são utilizadas no algoritmo AES, key_p e key_k. Esses valores são cifrados juntos com as chaves públicas geradas e são enviados para a RA.

Após a decifração do pacote, a RA é responsável por calcular efetivamente o valor de f_{kp} e f_{kc}. O algoritmo AES também necessita de valores de IV, que são obtidos com uma função aleatória segura ($os.urandom()$). Os valores f_k são transformados em objetos BigNumber big_fk_p e big_fk_k. A instanciação de BigNumber necessita da leitura de um binário com 32 bytes, que são obtidos a partir do valor de Hash de f_k. Depois disso, a RA multiplica big_fk_p pelo ponto gerador da curva e soma o valor resultante ao ponto $pseud$ recebido. O mesmo é feito para big_fk_c e $cypher$.

A RA realiza uma solicitação para a LA, informando o $request_id$ da solicitação, obtido através do Hash entre o $vehicle_id + pseud_point$. A RA também inclui na solicitação os valores $period_i$ e $j_certificates$, correspondentes ao período i e o número j de certificados válidos simultaneamente. Para este protótipo, utilizou-se o valor de i fixo em 1, mas esta

configuração pode ser alterada em modificações futuras. O valor de j pode ser alterado livremente e assumiu diversos valores durante os testes, como é discutido na seção 5.2.

Ambas as entidades LA calculam os $pre-linkagevalues$, ou valores pré-encadeamento, e os retornam em uma lista cifrada para a RA. O pacote elaborado pela LA corresponde a:

Código 4: Formato do pacote enviado pela LA (pré-cifração).

```
{
    'request_id',
    'period_i',
    'pre_linkage_values': [{
            'cyphertext',  # Correspondentes a
               pre_linkage_value_1
            'recovery_point'
        }, ..., {
            'cyphertext',  # Correspondentes a
               pre_linkage_value_j
            'recovery_point'
        }
    ]
}
```

Este pacote é cifrado com a chave pública da RA e enviado à entidade. Após o recebimento dos valores das LAs, inicia-se a montagem dos pacotes de envio para a PCA. As chaves casulos são expandidas em chaves lagartas e agrupadas com dois dos valores de pré-encadeamento, um de cada entidade LA. Os valores enviados pela RA são:

- *request_id*: Correspondente ao hash da concatenação entre vehicle_id e o binário da chave lagarta;
- *pseud_point*: Chave lagarta do pseudônimo;
- *cypher_point*: Chave lagarta de cifração;
- *la_package_1*: Valor de pré-encadeamento da LA_1, cifrado com a chave da PCA;
- *recovery_point_la_1*: Ponto para a recuperação do valor cifrado pela LA_1;
- *la_package_2*: Valor de pré-encadeamento da LA_2, cifrado com a chave da PCA;

- *recovery_point_la_2*: Ponto para a recuperação do valor cifrado pela LA_2.

A PCA expande as chaves lagartas em chaves borboletas e inclui os valores aleatórios utilizados na mensagem de retorno. São incluídos os campos $sign_1$ e $sign_2$, correspondentes à assinatura do pseudônimo, além do valor aleatório semente *seed*. O pacote é encriptado com a chave lagarta de cifração e enviado de volta à RA.

A RA junta todos os pseudônimos pertencentes ao mesmo veículo e coloca-os em uma lista. Junta-se também o valor IV utilizado no algoritmo AES para cada cifração. Esses dados são cifrados com a chave *cypher* casulo original e enviados de volta ao veículo. O veículo decifra o pacote com a chave privada *cypher* e deve calcular as expansões de chaves. Após a expansão das chaves casulos em chaves lagartas, é possível decifrar o pacote da PCA contendo o pseudônimo. A chave privada correspondente é calculada em sequência.

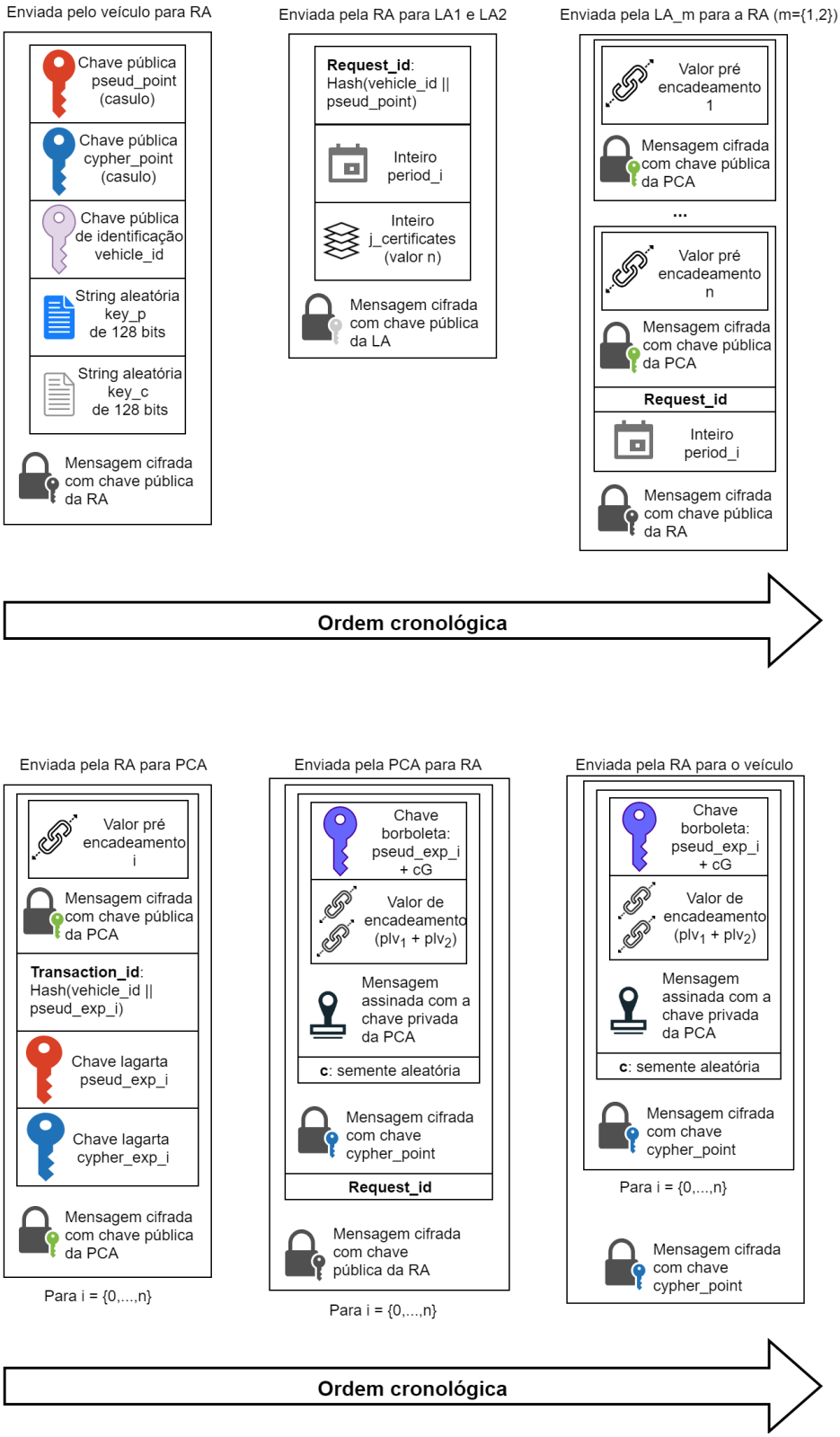
A figura 18 mostra o formato das mensagens enviadas pelas entidades. As mensagens estão organizadas em ordem cronológica de envio pelo sistema. Os dados obtidos a partir das transações são exibidos em uma interface web para cada uma das entidades. A figura 19 exibe a interface web correspondente às entidades LA.

5.2 Testes realizados

A lógica principal deste projeto baseia-se no uso da biblioteca criptográfica RELIC, que é acessada através da interface *pyrelic*. A primeira etapa de testes, portanto, refere-se ao correto funcionamento desta interface. Os resultados obtidos ao utilizar a biblioteca em um código C devem ser os mesmos ao compararmos com o código Python, dado que as entradas são iguais. Assim, foram elaborados testes unitários para validar as saídas produzidas pelo encapsulamento. A montagem dos testes foi baseada na escrita de um código C que executava as principais funções da biblioteca. Os parâmetros de entrada foram gerados aleatoriamente e extraídos para o código Python, juntamente com suas respectivas saídas. Em Python, as funções do módulo *pyrelic* foram chamadas e os resultados obtidos foram comparados. A cobertura dos testes abrange todas as classes do módulo, entre elas BigNumber, Point e CryptoProtocols.

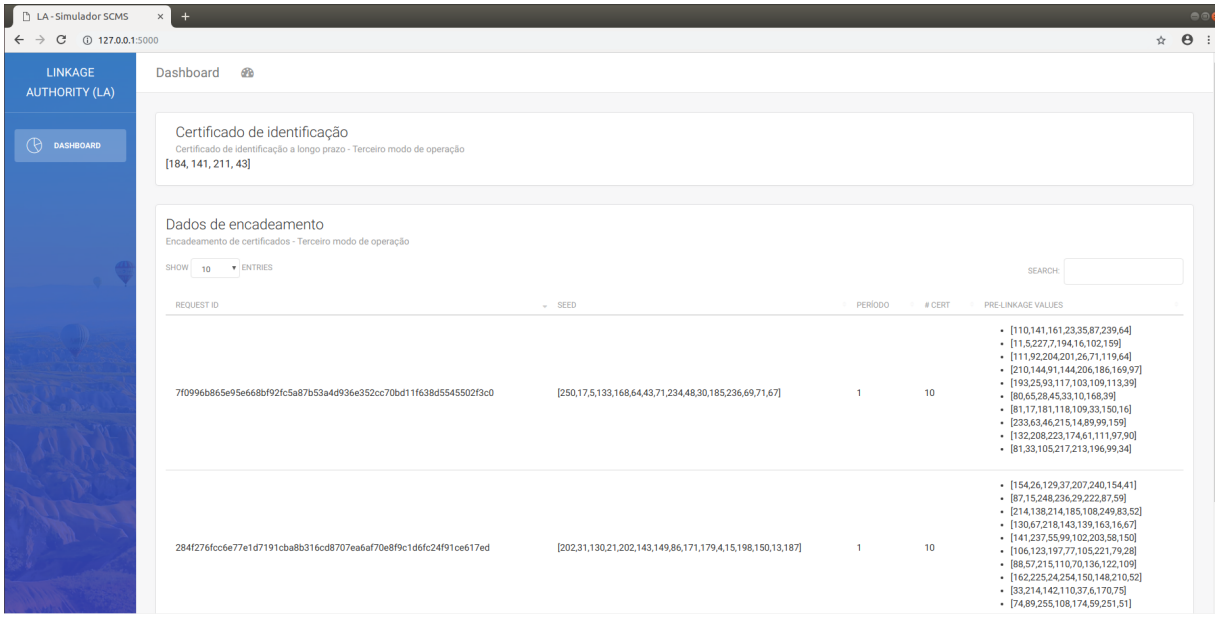
No projeto principal, deseja-se obter o tempo de execução da emissão de certificados para o veículo. Pode-se medir o tempo total da operação, assim como o tempo destinado à comunicação com a entidade do sistema. Esta diferenciação permite obter a porcentagem do processamento destinado ao veículo e a porcentagem em que ele está ocioso, esperando a

Figura 18: Formato das mensagens enviadas no terceiro modo de operação



Fonte: Autoria própria

Figura 19: Interface web para a LA



Fonte: Autoria própria

resposta da entidade. A meta para o terceiro modo de operação é obter uma porcentagem destinada ao veículo menor do que comparada ao segundo modo de operação.

Outra métrica interessante para medição é o tamanho das mensagens enviadas e recebidas pelo veículo. Deseja-se reduzir o consumo de banda do veículo, já que não é possível garantir a constância das redes de comunicação em cenários veiculares. Também é desejável que a solicitação feita pelo veículo tenha um tamanho uniforme e reduzido.

Neste contexto, os testes realizados obtiveram o tempo de execução no início da rotina de solicitação de certificados e ao final. Foram obtidos os tempos de execução antes e depois do envio de solicitação à entidade responsável. Também mediu-se o tamanho da mensagem enviada e recebida em bytes, através da biblioteca nativa "pickle"em Python. Esta biblioteca permite a conversão completa de um dicionário em uma string, cujo tamanho é então medido.

Este teste foi aplicado ao sistema veículo nos três modos de operação. Variou-se então a quantidade de certificados emitidos para a obtenção de resultados precisos. Para o primeiro e segundo modo de operação, realizou-se n solicitações para a obtenção de n certificados. Para o terceiro modo de operação, realizou-se uma única solicitação para obter n certificados, sendo que o valor n é ajustado no arquivo de configurações.

Testes foram realizados para 1, 10, 100, 500, 1000 e 2000 certificados, em cada um

dos modos de operação. Devido a limitações da máquina hospedeira, não foram realizados testes com parâmetros maiores, mas a extrapolação dos valores obtidos permite a previsão do comportamento do sistema. Os testes foram realizados em uma máquina com processador Intel i5-5200U, 8GB de RAM, executando uma máquina virtual Ubuntu 17.

5.3 Resultados obtidos

A partir dos testes, obteve-se o tempo de execução para os três modos de operação. A tabela 3 contém os valores obtidos para o primeiro modo de operação. Obteve-se o tempo médio de comunicação (TMC), que representa o intervalo médio em milissegundos gasto entre à solicitação do pseudônimo e sua recepção. O tempo médio de execução representa o intervalo médio em milissegundos de todo o procedimento de obtenção do certificado, desde a montagem do pacote pelo veículo até o armazenamento do pseudônimo no banco de dados.

Tabela 3: Resultados obtidos nos testes para o primeiro modo de operação

N	**Tempo médio comunicação (ms)**	**Tempo médio execução (ms)**	**% processamento no veiculo**
1	46,000	67,000	31,343
10	48,600	71,300	31,837
100	52,080	75,970	31,446
500	49,336	72,336	31,796
1000	49,733	72,962	31,837
2000	49,900	73,595	32,197

Fonte: Autoria própria

O primeiro modo de operação apresenta os menores valores para TMC e TME, devido à simplicidade da operação proposta e ao reduzido número de entidades. Observa-se também a constância dos tempos obtidos, mesmo ao aumentar o número de certificados gerados. Além disso, calculou-se o tempo gasto pelo veículo no processamento da transação, através da fórmula:

$$Processamento = \frac{TME - TMC}{TME} * 100 \tag{5.2}$$

Assim, o tempo gasto com processamento no veículo é da ordem de 30%. Este valor é alto, ao considerarmos que o dispositivo do veículo possui baixo poder computacional.

Na tabela 4, são exibidos os resultados dos testes para o segundo modo de operação. Observa-se que, mesmo nos valores mais baixos, os tempos obtidos são, no mínimo, o

dobro do tempo do primeiro modo de operação. Sabe-se que requisitos de segurança conflitam com requisitos de desempenho (STALLINGS, 2003) e isto é evidenciado pelos resultados do teste. O aumento no tempo médio é consequência direta do aumento de segurança e da garantia de privacidade fornecida pelo sistema.

Tabela 4: Resultados obtidos nos testes para o segundo modo de operação

N	**Tempo médio comunicação (ms)**	**Tempo médio execução (ms)**	**% processamento no veiculo**
1	94,000	134,000	29,851
10	115,700	156,000	25,833
100	125,530	170,500	26,375
500	146,796	197,774	25,776
1000	149,539	203,577	26,544
2000	151,219	206,889	26,908

Fonte: Autoria própria

Observa-se que a porcentagem de processamento realizado no veículo permanece entre 25% e 29%, representando uma ligeira melhora em relação ao modo anterior. Os dados da tabela permitem concluir que há um gargalo de desempenho na comunicação, uma vez que os valores de processamento mantêm-se próximos, mas os tempos médios crescem juntamente com o número de certificados.

Os resultados do terceiro modo de operação são exibidos na tabela 5. Observa-se um comportamento oposto ao do segundo modo de operação, no sentido em que baixas quantidades de certificados possuem altos valores de TMC e TME. À medida em que a variável N aumenta, os tempos médios de comunicação e execução reduzem, assim como a porcentagem de processamento realizado no veículo. Este comportamento é observado até $N = 1000$, valor no qual os tempos médios voltam a crescer, mesmo que em menor proporção.

A partir dos dados, observa-se que, para $N = 1$, o custo adicional gerado pelas entidades novas e cálculos extras não compensa, pois o sistema foi desenvolvido para a emissão em lotes. A medida em que aumenta-se o tamanho do lote, o custo adicional é diluído, a ponto de os valores de TMC e TME serem menores do que no segundo modo de operação, com $N = 500$. Esta é a faixa de operação ideal do sistema desenvolvido, uma vez que são emitidos o mesmo número de certificados do que no segundo modo de operação, mas o tempo médio de execução e o processamento no veículo são menores. Deve-se observar que, apesar da necessidade de bom desempenho computacional em sistemas veiculares, este protótipo não foi desenvolvido com esta prioridade. Assim, a faixa de

Tabela 5: Resultados obtidos nos testes para o terceiro modo de operação

N	**Tempo médio comunicação (ms)**	**Tempo médio execução (ms)**	**% processamento no veiculo**
1	458,000	509,000	10,020
10	195,900	219,500	10,752
100	163,910	180,560	9,221
500	174,400	190,974	8,679
1000	201,125	216,834	7,245
2000	211,748	227,212	6,806

Fonte: Autoria própria

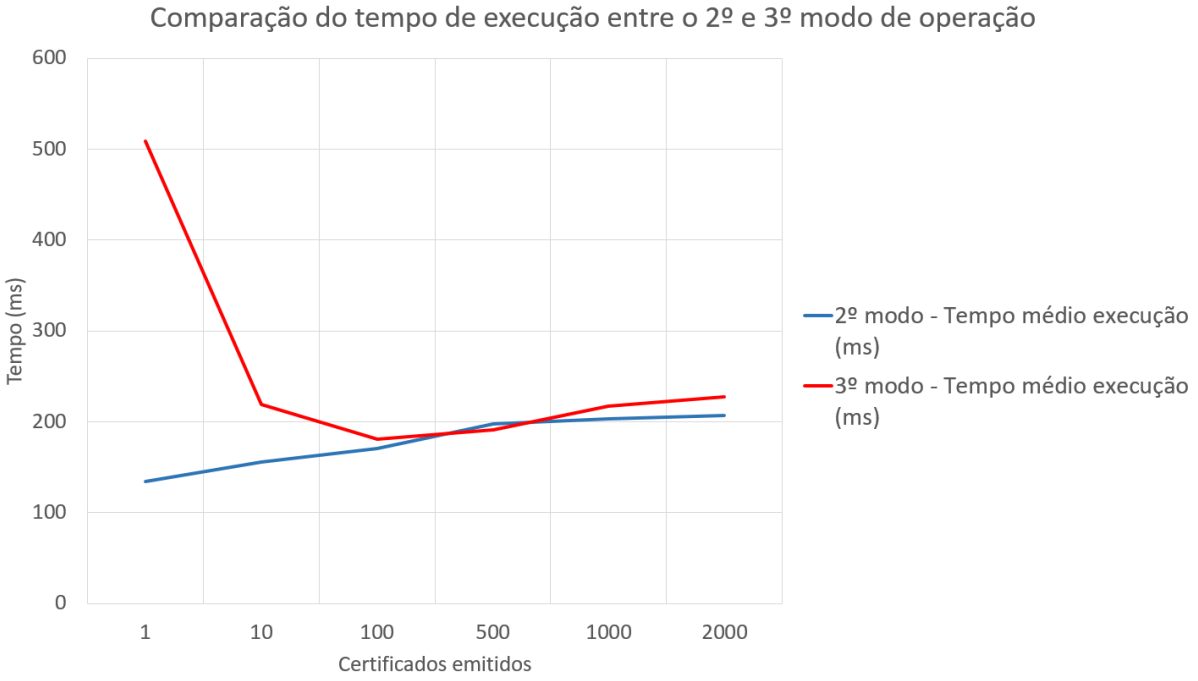
operação apenas aplica-se a este modelo.

A partir de $N = 1000$, os custos de comunicação tornam-se maiores do que as vantagens oferecidas pela solicitação em lote e expansão de chaves borboletas. Observa-se um aumento nos valores de TMC e TME, mas uma constante queda no processamento realizado pelo veículo. Em um sistema real, as entidades seriam executadas em servidores de alto desempenho, o que reduziria o tempo médio de comunicação e, consequentemente, o tempo médio de execução. Assim, a baixa porcentagem de processamento realizada no veículo garante a melhoria do desempenho ao executar o sistema num ambiente próximo ao real.

A tabela 6 exibe a quantidade de dados enviados e recebidos pelo veículo. Os dados consumidos no primeiro e segundo modo de operação são diretamente proporcionais ao número de certificados emitidos. Já no terceiro modo de operação, o tamanho da mensagem enviada é sempre constante e os dados recebidos aumentam proporcionalmente ao número de certificados. O terceiro modo de operação torna mais fácil a solicitação dos certificados, por enviar uma única mensagem, mas as mensagem recebidas possuem mais dados, uma vez que os certificados possuem valores de encadeamento e sementes aleatórias.

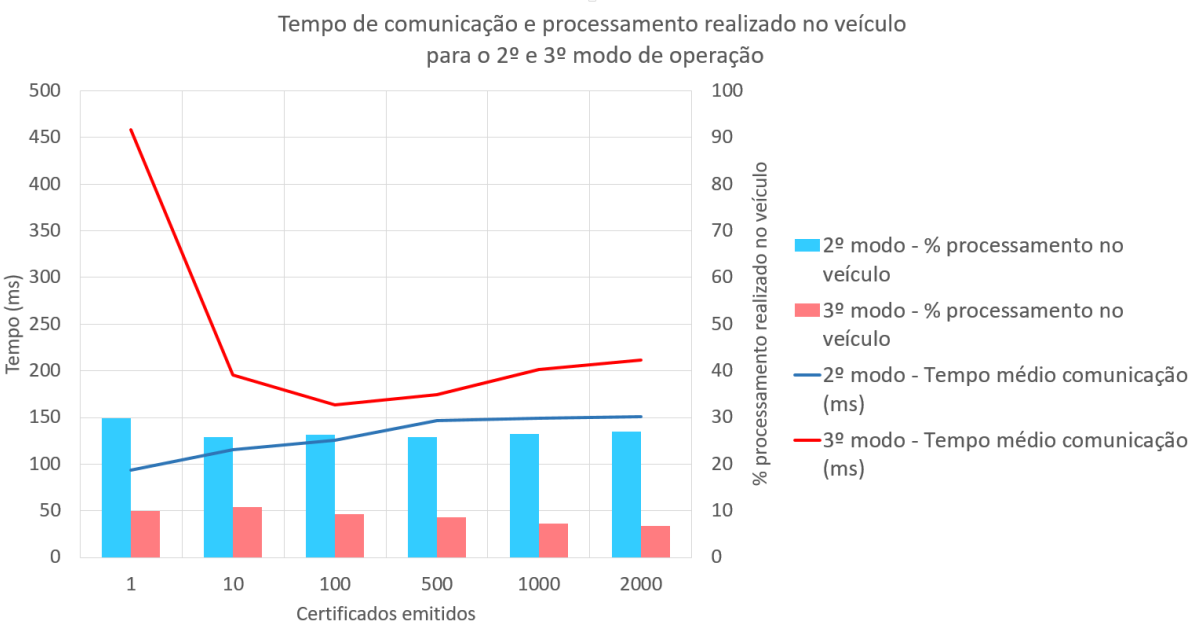
Apesar do tempo de execução ser maior para $N \geq 1000$, o terceiro modo de operação ainda é vantajoso por reduzir o tempo de processamento do veículo e por garantir mais segurança e privacidade ao usuário. Além disso, permite o encadeamento de pseudônimos, o que facilita a revogação de certificados maliciosos. Por fim, os gráficos 20 e 21 exibem a comparação visual entre os valores obtidos para o segundo e terceiro modo de operação.

Figura 20: Gráfico para o tempo de execução do segundo e terceiro modo de operação



Fonte: Autoria própria

Figura 21: Gráfico para o tempo de comunicação e porcentagem de processamento do segundo e terceiro modo de operação



Fonte: Autoria própria

Tabela 6: Resultados obtidos para consumo de banda

	Primeiro modo de operação		**Segundo modo de operação**		**Terceiro modo de operação**	
N	**Quantidade de dados enviados (kB)**	**Quantidade de dados recebidos (kB)**	**Quantidade de dados enviados (kB)**	**Quantidade de dados recebidos (kB)**	**Quantidade de dados enviados (kB)**	**Quantidade de dados recebidos (kB)**
1	1,210	0,858	1,215	1,146	1,530	6,592
10	12,131	8,580	12,109	11,703	1,530	63,224
100	121,003	85,802	121,766	119,427	1,530	627,372
500	605,015	429,011	607,863	576,467	1,530	3130,928
1000	1210,031	858,022	1215,348	1210,776	1,530	6265,260
2000	2420,062	1716,044	2442,877	2299,821	1,530	12535,268

Fonte: Autoria própria

6 CONSIDERAÇÕES FINAIS

Este capítulo final aborda as conclusões do projeto de formatura e as contribuições oferecidas por este trabalho. Também são discutidas as perspectivas de continuidade.

6.1 Conclusões do Projeto de Formatura

A comunicação veicular é parte fundamental das aplicações de segurança voltadas para carros autônomos e semi-autônomos. Através dela, é possível mapear os eventos que ocorrem na via e tomar medidas preventivas ou reativas, quando necessário. Os requisitos que devem ser atendidos relacionam-se à privacidade do usuário, à irretratabilidade de suas ações, à autenticação dos membros da rede, assim como à integridade das informações transmitidas. Nesse contexto, assinaturas digitais e pseudônimos satisfazem todas as exigências relativas à troca de mensagens entre veículos.

Dentre as soluções que utilizam pseudônimos, as mais promissoras empregam modelos de criptografia assimétrica e PKI, como é o caso do estudo SCMS. O sistema proposto é robusto, resistente à ataques internos e externos e preocupa-se com escalabilidade e desempenho no produto final. O advento de um modelo padrão de comunicação permitirá a popularização da tecnologia e a consequente instituição de sua obrigatoriedade entre montadoras de veículos, acelerando ainda mais o desenvolvimento e propagação de carros autônomos.

O protótipo desenvolvido procurou expor os pontos fortes do sistema SCMS e discutir sua superioridade frente a outras soluções atuais. O objetivo prático deste projeto foi concluído, uma vez que o sistema consegue demonstrar a importância de cada uma das operações do modelo original. Quanto aos modos de operação, conseguiu-se demonstrar as vantagens do terceiro modo, mesmo em cenários no qual não era o modo com execução mais rápida.

6.2 Contribuições

As contribuições deste trabalho podem ser divididas em aspectos conceituais e práticos. Com relação ao aspecto conceitual, este trabalho fez um amplo levantamento do estado da arte para a emissão de certificados em redes veiculares, permitindo ao leitor um conhecimento geral da área. Apesar de existirem artigos com enfoque no estado da arte (conhecidos como *surveys*), este documento possui uma coletânea de referências diferentes das encontradas, dando ênfase às soluções com criptografia assimétrica. Mesmo que não tenha-se produzido conhecimento novo, o compilado de estudos apresentados cria uma base sólida para sua produção durante um período de mestrado ou doutorado.

Com relação a aspectos práticos, este trabalho realizou a implementação de um protótipo para emissão de certificados em redes veiculares, utilizando como base o sistema SCMS. Sabe-se da existência de um simulador do sistema SCMS (VSC5, 2018), mas seu acesso é feito apenas através de uma interface de programação de aplicativos (*Application Program Interface* - API). Não é possível saber detalhes da implementação realizada ou de seu funcionamento interno, além de não ser possível propor alterações ao seu funcionamento. Desta forma, o presente trabalho disponibiliza o código da implementação do sistema, dadas as simplificações realizadas.

Por fim, outra contribuição prática refere-se ao encapsulamento da biblioteca RELIC para sua utilização em linguagem Python. Até a data de entrega deste trabalho, não foram encontrados projetos com este objetivo na Internet. O projeto *pyrelic* pode ser disponibilizado para uso geral pela comunidade desenvolvedora, desde que seja feita a a generalização de parâmetros e empacotamento do código em um módulo instalável Python.

6.3 Perspectivas de continuidade

O sistema proposto neste trabalho possui simplificações e escopo reduzido. A continuidade deste trabalho pode ser feita através da aproximação das condições reais de execução. O embaralhamento de solicitações feito pela RA, por exemplo, foi removido do escopo deste projeto e, em trabalhos futuros, poderia ser implementado para aumentar a privacidade dos usuários. Além disso, diversas entidades do sistema SCMS não foram abordadas neste protótipo e poderiam ser incluídas em uma versão mais completa, como é o caso da entidade de má-conduta MA ou a entidade responsável por esconder os dados físicos do veículo (*Location Obscurer Proxy* - LOP). Também é possível escalar o sistema

com réplicas das entidades principais, permitindo aceitar mais requisições dos veículos.

Com relação à migração da biblioteca RELIC para Python, uma proposta interessante de continuidade é a modularização do código desenvolvido. Através do empacotamento do módulo, é possível disponibilizá-lo para uso geral pela comunidade de desenvolvedores, fornecendo, assim, uma biblioteca criptográfica com alta precisão sob seus parâmetros.

Por fim, este projeto foi desenvolvido tendo em mente a participação no programa de Pré-Mestrado em Engenharia da Computação. Com a entrada no programa de mestrado, um possível caminho de continuidade é a proposta de melhorias ao sistema SCMS, para torná-lo mais seguro, robusto e escalável.

REFERÊNCIAS

BIMBRAW, K. Autonomous cars: Past, present and future a review of the developments in the last century, the present scenario and the expected future of autonomous vehicle technology. In: IEEE. *Informatics in Control, Automation and Robotics (ICINCO), 2015 12th International Conference on*. [S.l.], 2015. v. 1, p. 191–198.

FAGNANT, D. J.; KOCKELMAN, K. Preparing a nation for autonomous vehicles: opportunities, barriers and policy recommendations. *Transportation Research Part A: Policy and Practice*, Elsevier, v. 77, p. 167–181, 2015.

COMMITTEE, D. et al. Dedicated short range communications (dsrc) message set dictionary. *SAE Standard J*, v. 2735, p. 2015, 2009.

PETIT, J. et al. Pseudonym schemes in vehicular networks: A survey. *IEEE communications surveys & tutorials*, IEEE, v. 17, n. 1, p. 228–255, 2015.

WHYTE, W. et al. A security credential management system for v2v communications. In: IEEE. *Vehicular Networking Conference (VNC), 2013 IEEE*. [S.l.], 2013. p. 1–8.

STALLINGS, W. et al. *Computer security: principles and practice*. [S.l.]: Pearson Education, 2012.

DANG, Q. H. *Secure hash standard*. [S.l.], 2015.

WANG, X.; YIN, Y. L.; YU, H. Finding collisions in the full sha-1. In: SPRINGER. *Annual international cryptology conference*. [S.l.], 2005. p. 17–36.

BERTONI, G. et al. Keccak sponge function family main document. *Submission to NIST (Round 2)*, Citeseer, v. 3, n. 30, 2009.

DWORKIN, M. J. *SHA-3 standard: Permutation-based hash and extendable-output functions*. [S.l.], 2015.

STALLINGS, W. *Cryptography and network security: principles and practice*. [S.l.]: Pearson Education India, 2003.

DAEMEN, J.; RIJMEN, V. Aes proposal: Rijndael. 1999.

WILLIAM, S. *Cryptography and network security: principles and practices*. [S.l.]: Pearson Education India, 2006.

RIVEST, R. L.; SHAMIR, A.; ADLEMAN, L. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, ACM, v. 21, n. 2, p. 120–126, 1978.

HANKERSON, D.; MENEZES, A. J.; VANSTONE, S. *Guide to elliptic curve cryptography*. [S.l.]: Springer Science & Business Media, 2006.

CADILLAC. *V2V Safety Technology Now Standard on Cadillac CTS Sedans*. 2017. ⟨http://media.cadillac.com/media/us/en/cadillac/news.detail.html/content/Pages/news/us/en/2017/mar/0309-v2v.html⟩. (Accessed on 04/28/2018).

MERCEDES-BENZ. *E-Class Sedan — Mercedes-Benz*. 2017. ⟨https://www.mbusa.com/mercedes/vehicles/class/class-E/bodystyle-SDN⟩. (Accessed on 04/29/2018).

EDELSTEIN, S. *Volkswagen Jumps on the V2V Bandwagon, Plans to Launch System in 2019 - The Drive*. 2017. ⟨http://www.thedrive.com/tech/11920/volkswagen-jumps-on-the-v2v-bandwagon-plans-to-launch-system-in-2019⟩. (Accessed on 04/28/2018).

ABUELSAMID, S. *Toyota Has Big Plans To Get Cars Talking To Each Other And Infrastructure In the U.S*. 2018. ⟨https://www.forbes.com/sites/samabuelsamid/2018/04/16/toyota-launches-aggressive-v2x-communications-roll-out-from-2021/#209e3c56146c⟩. (Accessed on 04/28/2018).

SCHAUB, F.; MA, Z.; KARGL, F. Privacy requirements in vehicular communication systems. In: IEEE. *Computational Science and Engineering, 2009. CSE'09. International Conference on*. [S.l.], 2009. v. 3, p. 139–145.

SHAMIR, A. Identity-based cryptosystems and signature schemes. In: SPRINGER. *Workshop on the theory and application of cryptographic techniques*. [S.l.], 1984. p. 47–53.

KAMAT, P.; BALIGA, A.; TRAPPE, W. An identity-based security framework for vanets. In: ACM. *Proceedings of the 3rd international workshop on Vehicular ad hoc networks*. [S.l.], 2006. p. 94–95.

HE, D. et al. An efficient identity-based conditional privacy-preserving authentication scheme for vehicular ad hoc networks. *IEEE Transactions on Information Forensics and Security*, IEEE, v. 10, n. 12, p. 2681–2691, 2015.

CHAUM, D.; HEYST, E. V. Group signatures. In: SPRINGER. *Workshop on the Theory and Application of of Cryptographic Techniques*. [S.l.], 1991. p. 257–265.

CALANDRIELLO, G. et al. Efficient and robust pseudonymous authentication in vanet. In: ACM. *Proceedings of the fourth ACM international workshop on Vehicular ad hoc networks*. [S.l.], 2007. p. 19–28.

ZHANG, L. et al. A scalable robust authentication protocol for secure vehicular communications. *IEEE Transactions on vehicular Technology*, IEEE, v. 59, n. 4, p. 1606–1617, 2010.

PERRIG, A. et al. The tesla broadcast authentication protocol. *Rsa Cryptobytes*, RSA, v. 5, 2005.

FÖRSTER, D.; KARGL, F.; LÖHR, H. Puca: A pseudonym scheme with user-controlled anonymity for vehicular ad-hoc networks (vanet). In: IEEE. *Vehicular Networking Conference (VNC), 2014 IEEE*. [S.l.], 2014. p. 25–32.

KUMAR, V.; PETIT, J.; WHYTE, W. Binary hash tree based certificate access management for connected vehicles. In: ACM. *Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. [S.l.], 2017. p. 145–155.

DOUCEUR, J. R. The sybil attack. In: SPRINGER. *International workshop on peer-to-peer systems*. [S.l.], 2002. p. 251–260.

JR, M. A. S. et al. *A privacy-preserving method for temporarily linking/revoking pseudonym certificates in vehicular networks*. 2018. Cryptology ePrint Archive, Report 2018/185. ⟨https://eprint.iacr.org/2018/185⟩.

BRECHT, B. et al. A security credential management system for v2x communications. *IEEE Transactions on Intelligent Transportation Systems*, IEEE, p. 1–22, 2018. ISSN 1524-9050.

KISSNER, L.; SONG, D. Privacy-preserving set operations. In: SPRINGER. *Annual International Cryptology Conference*. [S.l.], 2005. p. 241–257.

ARANHA, D. F.; GOUVêA, C. P. L. *RELIC is an Efficient LIbrary for Cryptography*. 2009. ⟨https://github.com/relic-toolkit/relic⟩.

PYTHON. *3.6.5 Documentation*. 2001. ⟨https://docs.python.org/3/⟩. (Accessed on 04/14/2018).

DJANGO. *Django documentation — Django documentation — Django*. 2005. ⟨https://docs.djangoproject.com/en/2.0/⟩. (Accessed on 04/14/2018).

FOUNDATION, D. S. *Django Channels — Channels 2.1.5 documentation*. 2018. ⟨https://channels.readthedocs.io/en/latest/⟩. (Accessed on 11/11/2018).

WEBSOCKET-CLIENT. *Websocket-client: websocket client for python*. 2018. ⟨https://github.com/websocket-client/websocket-client⟩. (Accessed on 11/11/2018).

POSTGRESQL. *PostgreSQL: The world's most advanced open source database*. 1996. ⟨https://www.postgresql.org/⟩. (Accessed on 04/14/2018).

FOUNDATION, P. S. *ctypes — A foreign function library for Python — Python 3.7.0 documentation*. 2018. ⟨https://docs.python.org/3/library/ctypes.html⟩. (Accessed on 09/18/2018).

UBUNTU, C. L. *Ubuntu — The leading operating system for PCs, IoT devices, servers and the cloud*. 2018. ⟨https://www.ubuntu.com/⟩. (Accessed on 09/18/2018).

KITWARE. *CMake*. 2018. ⟨https://cmake.org/⟩. (Accessed on 09/18/2018).

VSC5, C. A. M. P. L. C. L. V. S. C. . *CycurV2X-SCMS Simulator*. 2018. ⟨https://scms.trustpoint.ca/⟩. (Accessed on 11/24/2018).