Creación de un dataset para Redes Ad-Hoc Vehiculares (VANETs) a través de ataques cibernéticos simulados

Autor

Dargenson David Aldana Prieto

Director

Luz Marina Santos Jaimes

Doctora en Ciencias de la Computación y Matemáticas Computacional

Universidad De Pamplona

Facultad De Ingeniería Y Arquitectura

Ingeniería De Sistemas

Pamplona - Colombia

2019

Agradecimientos

Quiero agradecer primeramente a Dios por guiarme en el camino y fortalecerme espiritualmente para emprender el camino que me guía a la realización de mis sueños.

Agradecer a mi madre quien me dio la vida y a mi familia que me enseño que la perseverancia y la constancia es el pilar del éxito, en especial a mi madre Ereida Prieto y a mis tíos Norberto Flórez y María Carlina Flórez quienes siempre han creído en mí, que con sus consejos y su apoyo incondicional me empuja día a día para que logre mis metas. Así mismo, a mis hermanos que con su apoyo y compañía siempre han estado cuando más los he necesitado.

Presento mi más sincero agradecimiento a mis tutores, quienes con su conocimiento, paciencia y guía fueron una pieza clave para el desarrollo de mi proyecto y todas aquellas personas que estuvieron presentes en la realización de este sueño que es tan importante para mí.

Tabla de contenido

Lista de figuras

LISTA TABLAS

Lista de siglas y acrónimos

ART	Umbral de rango de aceptación
AU	Application Unit
BSM	Basic Safety Message
CCH	Control Channel
CSMA/ CA	Carrier Sense Multiple Access/Collision Avoidance
dataset	Conjunto de datos
DDoS	Distributed Denial of Service
DMV	Verificación de la distancia recorrida
DoS	Disk Operating System
DSRC	Dedicated Short-Range Communication
FCC	Federal Communications Commission
GPS	Global Positioning System
I2I	Infrastructure to Infrastructure
IP	Internet Protocol
ITSA	Intelligent Transportation Society of America
IVC	Inter-Vehicle Communication
JSON	JavaScript Object Notation
MAC	Medium Access Control
OBU	On-Board Unit
OFDM	Orthogonal Frequency Division Multiplexing
OSI	Open System Interconnection
PES	Presence Evidence System
RM	Resource Manager
RSU	Road-Side Unit
SAW	la advertencia de aparición repentina
SCH	Service Channel
SSC	la verificación de velocidad simple
TCP	Transmission Control Protocol
TFP	Task Force p
TRACI	Traffic Control Interface
UDP	User Datagram Protocol
V2I	Vehicle To Infrastructure
V2V	Vehicle To Vehicle
VANETs	Redes Ad-hoc Vehiculares
WAVE	Wireless Access in the Vehicular Environment
WBSS	WAVE Basic Service Set
WSA	WAVE Service Advertisement

Creación de un dataset para Redes Ad-Hoc Vehiculares (VANETs) a través de ataques cibernéticos simulados

Resumen:

Las Redes Ad-hoc Vehiculares (VANETs) son redes de vehículos inteligentes, una importante tecnología que ofrece una amplia gama de servicios, que pueden estar relacionados con la seguridad, la optimización y el entretenimiento. Estas comunicaciones se pueden utilizar para desviar el tráfico de rutas congestionadas, para ayudas de emergencia y para prevenir accidentes, además de proporcionar un control de tráfico inteligente, entre otras. Sin embargo, la seguridad de las VANETs se convierte en un desafío crítico, ya que un atacante puede usar el mismo sistema para difundir mensajes falsos que causan congestión de tráfico y accidentes viales.

Este trabajo investiga la aplicación de la detección de mala conducta en escenarios simulados donde un atacante interfiere con la comunicación en las VANETs, se toma como base un proyecto del estado del arte donde se estudia los procedimientos y la codificación del ataque de falsa posición. Con lo anterior, se da partida a la creación de un código abierto donde se abordan los ataques *Sybil* y *Timing*, al simular estos ataques se obtiene un conjunto de datos (dataset) que se utilizan para análisis de los datos. Posteriormente se valida el dataset de cada ataque mediante la detección de plausibilidad utilizando mecanismos de detección que con el cálculo de precisión y recuperación se puede determinar cuál es el más viable para la detección de los ataques simulados. Con los resultados se formula una documentación detallada del dataset para mejor interpretación en futuros estudios e investigaciones.

Palabras claves:

Redes Ad-Hoc Vehiculares, detección de mal comportamiento, ataque *Sybil*, ataque *timing*, dataset, aplicaciones de seguridad.

Abstract:

Ad-hoc Vehicular Networks (VANETs) are intelligent vehicle networks, an important technology that offers a wide range of services, which may be related to safety, optimization and entertainment. These communications can be used to divert traffic from congested routes, for emergency aids and to prevent accidents, in addition to provide intelligent traffic control, among others. However, the security of VANETs becomes a critical challenge, since an attacker can use the same system to spread false messages that cause traffic congestion and road accidents.

This document investigates the application of the detection of misconduct in simulated scenarios where an attacker interferes with the communication in the VANETs, it is based on a project of the state of art where the procedures and coding of the false position attack were studied. To do this, start the creation of an open source where *Sybil* and *Timing* attacks are addressed, by simulating these attacks you get a set of data (dataset) that will be used for data analysis, then the dataset is validated. each attack by means of the plausibility detection using detection mechanisms that with the calculation of precision and recovery can determine which is the most viable for the detection of the simulated attacks. And with that formulate a detailed documentation of the dataset for better interpretation in future studies and research.

Keywords:

Vehicular Ad Hoc Network, Misbehavior Detection, Sybil Attack, Timing Attack, dataset, Safety App.

INTRODUCCIÓN

Las Redes Ad-hoc Vehiculares (VANETs) han recibido una gran atención en la comunidad de investigación en las últimas décadas como una tecnología que ofrece una amplia gama de servicios, que pueden estar relacionados con la seguridad, la optimización y el entretenimiento. Estas redes, que consisten en vehículos con módulos de comunicación inalámbrica ad-hoc, están ganando importancia en el contexto de las aplicaciones cooperativas de conducción autónoma. La idea es que la comunicación puede mejorar significativamente la conducción autónoma al aumentar esencialmente la disponibilidad de información dentro del vehículo. Sin embargo, para que estas aplicaciones funcionen correctamente, esta información debe ser autenticada y verificada para corrección (Heijden, Lukaseder, & Kargl, 2018). Para lograr que la información este correcta las agencias de estandarización han creado estándares criptográficos y de comunicación (IEEE 802.11p, IEEE 1609) donde se identificó que esta solución solo proporciona integridad de los datos y no garantiza la corrección de los mensajes (a esto se le denomina detección de mal comportamiento).

En estas redes hay variedad de desafíos de investigación pendientes en el área de detección de mal comportamiento la réplica para evaluar las aplicaciones VANETs, que se pueden clasificar en tres grupos: estudios de campo del mundo real, modelos analíticos y simulaciones (Heijden, Lukaseder, & Kargl, 2018). Este último estudio se utiliza a menudo como principal método de evaluación para VANETs y para la creación de conjuntos de datos. Ya que un conjunto de datos proporciona una base inicial para comparar los mecanismos de detección y reducir el tiempo requerido en las investigaciones.

Este documento adopta el enfoque de simulaciones en aplicaciones de seguridad, ya que permite simular ataques cibernéticos y dar como salida un conjunto de datos. Para este proyecto se implementan dos ataques cibernéticos: en primer lugar, el ataque *Sybil* que consiste en que un nodo atacante fabrica una serie de nodos fantasmas que busca que un nodo legítimo piense que hay muchos vehículos alrededor y tome una ruta predicha por el atacante (Joe & Ramakrishnan, 2016), (Sakiz & Sen, 2018) y (Yua, Xu, & Xiao, 2013). En segundo lugar, el ataque *Timing* que consiste en que un nodo accidentado manda un mensaje de emergencia a los nodos cercanos y uno de ellos, el atacante lo reenvía, pero retrasa el mensaje programando su envío un intervalo de tiempo más al tiempo actual, causando que los nodos cercanos a este al llegarle tarde el mensaje tomen la ruta del

accidente y se genere una congestión (Joe & Ramakrishnan, 2016) y (SUMRA, MANAN, & HASBULLAH, 2014). Al implementar estos ataques da como salida un conjunto de datos que posteriormente será utilizado para validarlo con mecanismos de plausibilidad y a su vez analizar gráficamente los resultados.

1. PROBLEMA

1.1. Planteamiento del problema

Teniendo en cuenta que las Redes Ad-hoc Vehiculares (VANETs) tienen como objetivo evitar congestiones de tráfico y accidentes viales a través de aplicaciones de seguridad y aplicaciones de optimización, se requiere mecanismos de seguridad y privacidad para protección de los mensajes en estas redes. Las VANETs por su naturaleza inalámbrica, topología dinámica y alta movilidad están sujetos a vulnerabilidades que son aprovechadas por vehículos maliciosos actuando como hackers donde utilizan ataques cibernéticos para modificar, guardar y eliminar información pertinente para la comunicación entre vehículos (Sakiz & Sen, 2018), (Heijden, Lukaseder, & Kargl, 2018) y (Le, Hartog, & Zannone, 2018).

Para verificar el mal comportamiento de los vehículos y que los datos obtenidos por medio de mensajes en VANETs son correctos, existe un número de variables que describen el comportamiento entre los datos y la detección de ataques. Lo cual representa un problema que requiere el uso de dataset que facilite el tratamiento, análisis y extracción de los datos que será relevante para mejorar los mecanismos de seguridad y privacidad en VANETs. De lo anterior surge el interrogante de la presente propuesta, ¿Es viable analizar el comportamiento de las VANETs ante un ataque cibernético a partir de los datos obtenidos de un escenario de simulaciones?

1.2. Justificación

En las últimas décadas, varias industrias automotrices han tomado medidas de seguridad, como las bolsas de aire, los sistemas de frenos antibloqueo y los sistemas electrónicos de control de estabilidad, los cuales se han desarrollado e implementado, mejorando la seguridad de los vehículos. Sin embargo, los choques siguen ocurriendo, causando muertes, lesiones y daños a la propiedad. En la actualidad, las tecnologías para evitar colisiones están atrayendo una atención significativa en la industria automotriz para disminuir e incluso evitar los accidentes por completo (Le, Hartog, & Zannone ,2018). Una tendencia emergente en la prevención de choques y accidentes es el desarrollo de las tecnologías VANETs que tienen como objetivo predecir y advertir a los conductores de accidentes inminentes. Lo cual impulsa el concepto de dataset, ya que este proporciona una base inicial para comparar los mecanismos de detección y reducir el tiempo requerido en las investigaciones de simulación de alta calidad que faciliten la comparación de resultados en diferentes proyectos. La presente propuesta está alineada a los intereses del proyecto VeReMi (Heijden, Lukaseder, & Kargl, 2018), con lo que se pretende extender la información ya establecida y dejar un documento explicando lo analizado en el conjunto de datos para posteriores estudios.

1.3. Delimitaciones

1.3.1. Objetivo general

- Generar un dataset en Redes Ad Hoc Vehiculares (VANETs) mediante ataques cibernéticos simulados.

1.3.2. Objetivos específicos

- Analizar los aspectos referentes a seguridad de las VANETs para comprensión de los ataques a simular.
- Codificar en las herramientas de simulación los ataques para la creación del dataset.
- Validar el dataset de ataques mediante un algoritmo de detección para obtención de resultados.

1.4. Metodología

En primer lugar, se estudiaron todos los temas relacionados con el proyecto (VANETs, ataque *Sybil*, ataque *Timing*, simuladores y codificación en lenguaje Python). Lo anterior con el fin de dominar los temas y hacer la planeación del proyecto. Luego se analizó la codificación y aplicación realizada en el proyecto VeReMi, se continuó con la instalación de los tres simuladores SUMO, OMNeT y VEINS, y corrección de errores para la ejecución de las simulaciones, de tal manera que se mostrará el comportamiento de las variables de salida y estructura que tienen los archivos json resultantes. A continuación, se programó el código necesario para implementar los ataques *sybil* y *Timing* y ejecutar el conjunto de simulaciones planeadas para la creación del dataset.

En segundo lugar, para la validación de dataset obtenido se hizo una selección de los proyectos y artículos estudiados en el estado del arte con el objetivo de determinar cuál validación sería la más adecuada para implementarla en el proyecto. De esta selección se escogió la del proyecto VeReMi por su facilidad de interpretación e implementación en

diferentes lenguajes de programación. Al tener esto presente se eligió el lenguaje Python para su desarrollo, ya que permite implementar algoritmos de manera más intuitiva con ayuda de sus librerías. Posteriormente, se estudió y analizó la validación donde se observó la utilización de mecanismos de plausibilidad para la detección de mala conducta de los mensajes. Por otro lado, estos mecanismos son aquellos que tienen algún modelo implícito o explícito del mundo real y verifican sí la información entrante es aceptable dentro de este modelo. Teniendo claro los conceptos y los mecanismos utilizados en la validación (ART, SAW, SSC, DMV que posteriormente serán explicados en las siguientes secciones), se hace una réplica de su validación en lenguaje Python utilizando la herramienta Anaconda. Al tener lista la réplica denominada validación 1, se modifica agregando unas reglas de detección para cada uno de los ataques planeados, a esta modificación se le llama validación 2. Cada una de estas validaciones evalúa los mensajes por nodo, permitiendo registrar en diferentes umbrales el comportamiento de cada mecanismo en un archivo CSV con el fin de graficar de manera adecuada este comportamiento. Al final se analizan los resultados de las validaciones.

1.5. Acotaciones

El presente trabajo se basó en el proyecto VeReMi (Heijden, Lukaseder, & Kargl, 2018), buscando la simulación de los ataques *Sybil* y *Timing*. El dataset generado por las simulaciones consistió en un conjunto de archivos con extensión *. json. Estos archivos contienen el registro de los mensajes (*beacons*) intercambiados entre los vehículos. Cada mensaje contiene principalmente los siguientes datos: identificación del vehículo, velocidad del vehículo, longitud, latitud, ángulo de rotación, tiempo de emisión y recepción de los mensajes. Se trabajaron tres categorías de densidad vehicular: Baja densidad correspondiente a una movilidad a las 3:00 hs que tiene de 180 a 192 vehículos, media densidad con una movilidad a las 5:00 que tiene entre 460 y 475 vehículos, y alta densidad con una movilidad de 7:00 que tiene entre 816 y 843 vehículos. El dataset sirve de entrada a un algoritmo de detección de mal comportamiento que se seleccionó del estado del arte.

2. MARCO TEÓRICO

2.1. Redes Ad-Hoc Vehiculares

En esta parte se incluye en detalle el concepto de Redes Ad-Hoc Vehiculares (también denominado en sus siglas en ingles VANETs), características, aplicaciones y entre otros subtemas que facilitan la comprensión del proyecto.

2.1.1. Definición.

Según (Orozco Sarasti y Llano Ramírez ,2015), "las VANETs son un caso particular de las MANETs (*Mobile Ad- hoc Networks*) que han surgido gracias a los avances tanto en las tecnologías inalámbricas e investigaciones en la industria automotriz para desarrollo de redes enfocadas a entornos vehiculares. Estas redes son un conjunto de nodos que se comunican entre sí mediante enlaces inalámbricos sin la necesidad de una infraestructura de red fija. Cada vehículo se define como un nodo de la red y está equipado con una unidad de comunicación a bordo denominada OBU (*On-Board Unit*) y una unidad de aplicación llamada AU (*Application Unit*). La función de la OBU es intercambiar información con otros vehículos o con puntos de acceso estacionarios ubicados alrededor de las carreteras, denominados RSU (*Road-Side Unit*), mientras que las AU hacen referencia a los dispositivos que muestran información al usuario. Generalmente se les da esta denominación a dispositivos como computadores portátiles, *smartphone* o pantallas ubicadas dentro del nodo que se encuentran conectados a la OBU y a su vez cada nodo actúa como *router* que tiene capacidades de encaminar paquetes.

Los elementos que conforman las VANETs al operar entre sí forman dominios, que hacen referencia a un conjunto de elementos lógicos y físicos que funcionan colectivamente para establecer comunicaciones entre nodos y la infraestructura. Estos dominios se clasifican de acuerdo a su funcionamiento en:

- Dominio en el vehículo: Conformado por la OBU y las AUs del nodo. lo cual, forman

una red de comunicación bidireccional dentro del vehículo y pueden conectarse alámbrica o inalámbricamente.

- Dominio Ad hoc: Este dominio hace referencia a la comunicación inalámbrica utilizada para enlazar los nodos entre sí o los nodos con las RSU. Dicha comunicación puede entablarse mediante el estándar presentado por la IEEE denominado IEEE 802.11p (junto con sus variantes específicas para seguridad, *networking*, gestión de recursos y operaciones multicanal) o mediante otras tecnologías inalámbricas.
- Dominio de infraestructura: Formado por las redes de acceso y la infraestructura que soporta el acceso a Internet que solicitan los nodos y/o las RSU ". Los componentes y dominios descritos anteriormente se muestran en la Figura 1.

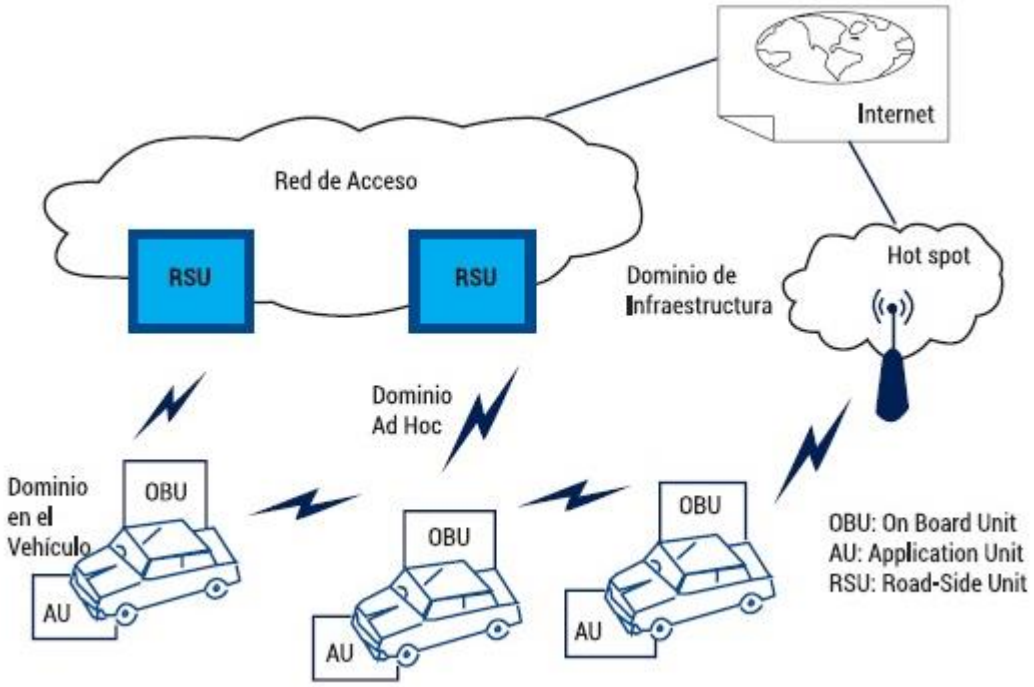

Figura 1. Dominios y componentes de una red vehicular.
Fuente: (Orozco, Chavarro & Calderón, 2013)

2.1.2. Características.

A continuación, el conjunto de características que definen estas redes (Garc y Guti ,2014):

- Autonomía: Cada nodo es autónomo con capacidad de procesar la información que se intercambia en la red. El control de la red no depende de una infraestructura externa, sino que se distribuye en todos los nodos de la red siendo así más tolerante a fallos.
- Encaminamiento distribuido: De la misma manera que son autónomos, los nodos deben ser capaces de encaminar información, deben tener capacidades de *router*. Por lo tanto, es necesario definir nuevos protocolos de encaminamiento capaces de soportar esa característica.
- Topología de red variable: En una MANET los nodos se pueden mover de forma arbitraria. Esa característica se debe matizar en el caso de las VANETs ya que los vehículos suelen seguir un cierto patrón de movimiento, por ejemplo, siguiendo las curvas de un circuito urbano. Aun así, los vehículos se mueven de forma más rápida que un terminal en una red móvil clásica. Debido a esa variabilidad de posición se pueden producir pérdidas importantes de paquetes. Serán necesarios mecanismos que detecten estas circunstancias y minimicen sus efectos.
- Capacidad variable de los enlaces: Esta característica tiene cabida en todas las comunicaciones inalámbricas, pues es intrínseca al medio de transmisión, pero sus efectos se agravan más en las VANETs. Esto se debe a que cada nodo actúa como *router* y la información atraviesa múltiples enlaces inalámbricos.
- Terminales limitados: En la mayoría de los casos los nodos de este tipo de redes serán terminales ligeros embarcados en vehículos con capacidades limitadas de procesamiento, comunicación y alimentación por lo que es primordial que los algoritmos utilizados optimicen estos tres recursos.

2.1.3. Tipos de comunicación

Dentro del entorno de una red vehicular se pueden producir varios tipos de comunicación entre los nodos. Podemos encontrar los siguientes (Sánchez, 2017):

- Comunicaciones de vehículo a vehículo (V2V): Comunicaciones efectuadas entre los vehículos de un modo ad-hoc espontáneo. En este tipo de comunicación un vehículo puede intercambiar información valiosa con otros vehículos como, por ejemplo, el estado de la vía mediante la detección y aviso de accidentes u obras, las condiciones actuales del tráfico u otros servicios susceptibles del uso de este tipo de comunicación.

- Comunicaciones de vehículo a infraestructura (V2I): Este tipo de comunicación se realiza entre los vehículos y las estaciones situadas al lado de la carretera (RSUs). Sirven, por ejemplo, para intercambiar información sobre las condiciones de la vía y avisos para que la infraestructura tenga en consideración la información acerca del entorno de la red y de los nodos que se comunican con ella. Este tipo de comunicaciones también se utiliza como enlace para el intercambio de datos con redes externas, por ejemplo, Internet.

- Comunicaciones de infraestructura a infraestructura (I2I): Este tipo de comunicación suele ser el menos habitual dentro de una red vehicular completa. Se encuentra útil la posibilidad del intercambio de información entre infraestructuras obteniendo beneficios como la difusión de información recopilada por los vehículos y enviada previamente a las infraestructuras para, posteriormente, ofrecer servicios de información al conductor. Este tipo de comunicación puede estar implementada haciendo uso de infraestructuras existentes como puede ser el cableado de conexión a Internet ubicado en las vías urbanas. En la Figura 2 se observan los tipos de comunicación nombrados anteriormente:

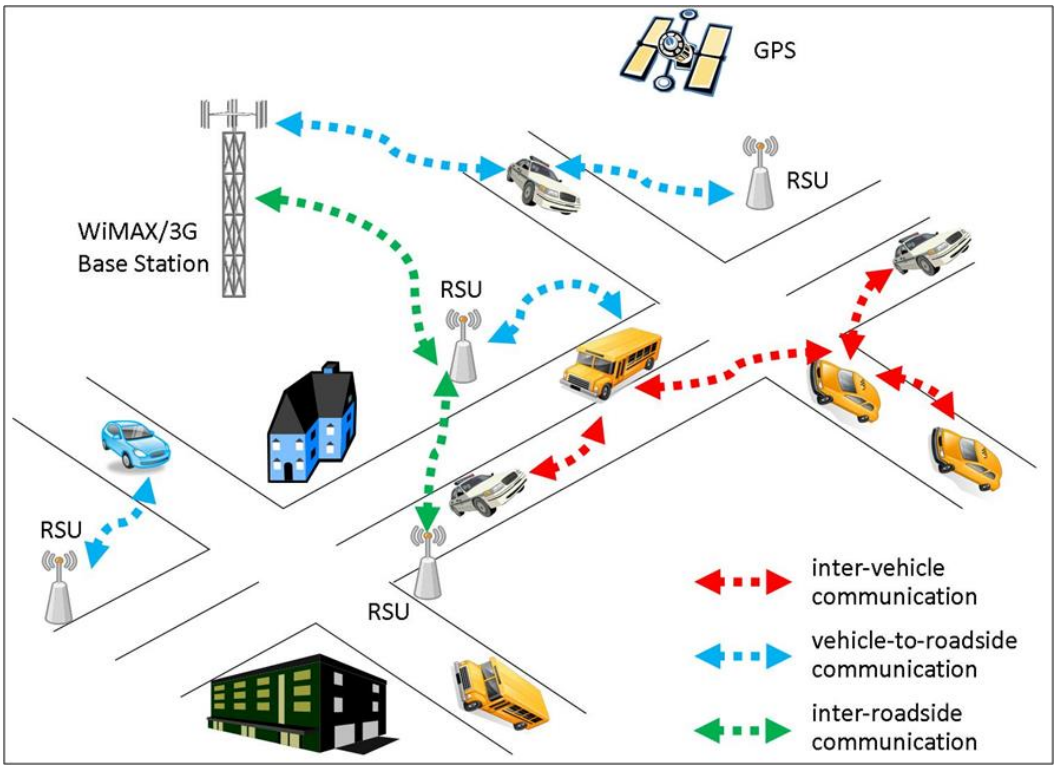

Figura 2. Tipos de comunicaciones en VANETs
Fuente: (reu2015.weebly, 2019)

2.1.4. Estándares (IEEE 802.11p, 1609)

La demanda de movilidad, de cobertura geográfica y el alto grado de variabilidad de las redes vehiculares son factores determinantes en la elección de los protocolos y de los estándares adecuados para la comunicación V2X (*vehicle to anything*).

En el año 1999 la FCC de Estados Unidos (*Federal Communications Commission*) asignó 75 MHz de ancho de banda en la frecuencia 5,9 GHz para los servicios del sistema inteligente de transporte denominado DSRC (*Dedicated Short-Range Communication*). Posteriormente, en el año 2002 la ITSA(*Intelligent Transportation Society of America*) recomendó la adopción de un único estándar para la capa física y para la capa MAC (*Medium Access Control*) propio de las VANETs. En consecuencia, en el año 2004 se consolidó el grupo de trabajo IEEE TFP (*Task Force p*) con el objetivo de especificar una arquitectura de comunicación para entornos vehiculares, basada en la tecnología inalámbrica para redes de área local IEEE 802.11, denominada IEEE 802.11p. Más adelante, el grupo de trabajo IEEE

1609 desarrolló un conjunto de especificaciones de las capas de red, transporte y aplicación para las comunicaciones vehiculares.

Los estándares IEEE 802.11p e IEEE 1609 definen el acceso inalámbrico en ambientes vehiculares, este conjunto de protocolos se denomina WAVE (*Wireless Access in the Vehicular Environment*). WAVE proporciona una arquitectura para las comunicaciones V2X, destinada al uso aplicaciones de seguridad y eficiencia vial. La Figura 3 muestra la pila de protocolos de la arquitectura WAVE siguiendo el modelo de referencia OSI (Orozco, Llano, y Michoud ,2016).

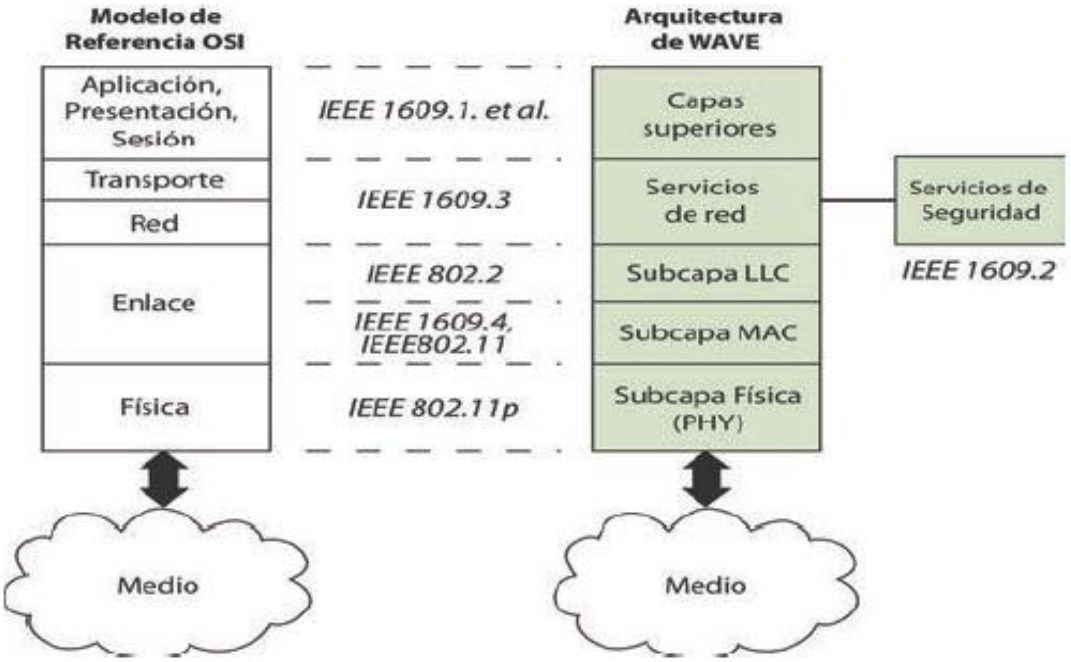

Figura 3. Estándares pertenecientes al modelo WAVE
Fuente: (Orozco, Llano, y Michoud ,2016)

El estándar IEEE 802.11p define las características de la capa física y MAC necesarias para las VANETs (Sedano & Gil, 2019) y (Sánchez, 2017).

- Capa física: Está basado en la capa física de 802.11a, es decir, implementa OFDM (*Orthogonal Frequency Division Multiplexing*). La Tabla 1 muestra las características asociadas al canal físico de IEEE 802.11p.

Características	Valor
Velocidad de transmisión	3-27 Mbps
Rango de comunicación	< 1000 m
Potencia máxima de transmisión	760 mW (US) 2 W (EU)
Ancho de banda del canal	10 MHz
Espectro de frecuencia asignado	75 MHz (US) 30 MHz (EU)
Bandas de frecuencia	5,86-5,92 GHz

Tabla 1. Información del canal inalámbrico V2X, IEEE 802.11p
Fuente: (Emmelmann, Bochow &Kellum, 2010)

- Capa MAC: El propósito de la capa MAC es establecer los mecanismos de acceso al canal de comunicación, de manera que un conjunto de estaciones pueda compartir eficientemente el medio inalámbrico. El estándar IEEE 802.11p define el uso de CSMA/ CA (*Carrier Sense Multiple Access/Collision Avoidance*) para las comunicaciones V2X. La capa MAC también considera aspectos de la transmisión como la probabilidad de recepción de paquetes, el tiempo de acceso al canal, el control de congestión y la priorización de los mensajes. En la arquitectura de comunicaciónV2X se establecen dos tipos de canal: canal de control y el canal de servicio:

    1. CCH (*Control Channel*): Es un canal de radio fijo, usa comunicación tipo *broadcast*, dedicado para paquetes de alta prioridad y baja latencia. Las aplicaciones de seguridad críticas hacen uso de él.

    2. SCH (*Service Channel*): Se establece para la comunicación en dos vías entre la OBU y la RSU. El canal SCH es usado por aplicaciones específicas y éstas pueden ser ejecutadas en paralelo en diferentes canales de servicio. Es necesario el establecimiento de un conjunto de servicios denominados WBSS (*WAVE Basic Service Set*) que establece la comunicación ad hoc entre las unidades OBU y RSU.

La familia de estándares IEEE 1609 define los aspectos de operación y gestión de la capa de red, de transporte y de aplicación de la arquitectura WAVE. A continuación, se describe brevemente cada estándar (Sarasti & Ramírez, 2014).

- **I**EEE 1609.0: Define la arquitectura general de WAVE, el modelo de comunicación, los mecanismos de acceso al medio inalámbrico en ambientes vehiculares, la estructura general de los componentes como OBU, RSU e interfaces WAVE.
- IEEE 1609.4: Describe la operación multicanal que implementa la capa física, incluyendo los parámetros para la priorización de mensajes, temporizadores, conmutación del canal y primitivas diseñadas para el funcionamiento multicanal.
- IEEE 1609.3: Describe los servicios de la capa de red para entornos vehiculares, especifica las funciones de enrutamiento y direccionamiento basados en el nivel 3 del modelo de referencia OSI y de los protocolos IP, UDP, TCP y WSMP.
- IEEE 1609.2: Especifica los servicios de seguridad en los sistemas WAVE, define los formatos de los mensajes y su procesamiento.
- IEEE 1609.1: Describe el RM (*Resource Manager*) en los sistemas WAVE, que permite a una OBU con capacidad de cómputo limitada ejecutar procesos de manera remota.

2.1.5. Aplicaciones

Uno de los principales objetivos de los ITS (*Intelligent Transportation Systems*) lo constituye el brindar un mejor escenario de conocimiento de las carreteras a los conductores, para de cierta forma poder reducir el número de accidentes y a su vez la conducción se pueda realizar de una manera más cómoda y fluida.

Para las VANETs, las aplicaciones pueden ir desde un simple intercambio de información entre sus nodos, hasta el poder tener acceso a contenidos Multimedia e Internet. Según (García & Águila, 2016), “las aplicaciones VANETs pueden ser de seguridad, optimización y entretenimiento.”

Las aplicaciones de seguridad componen una de las bases de la motivación del desarrollo de redes vehiculares ofreciendo a la comunidad nuevos sistemas de previsión de accidentes, información del tráfico en tiempo real, entre otras. Por otro lado, estas aplicaciones se comunican a través de mensajes BSMs (*Basic Safety Messages*) o WSAs (*WAVE Service Advertisement*) con el fin lograr su objetivo el cual es evitar congestiones y accidentes de tráfico. Este tipo de aplicaciones se pueden clasificar en (Sánchez, 2017):

- Aplicaciones de Tráfico en Tiempo Real: Este tipo de aplicaciones están destinadas al almacenamiento en tiempo real del tráfico en ciudad pudiendo identificar posibles atascos o zonas conflictivas por contaminación u otros agentes y compartiendo la información con el resto de nodos de la red.

- Aplicación de Transferencia de Mensajes Cooperativos: Este tipo de intercambio de mensajes están destinados a sistemas cooperativos que permiten ayudar a otros vehículos.

- Aplicaciones de Notificación de Accidentes: Este tipo de aplicaciones otorgan a los usuarios de la red una consciencia plena sobre el estado de la vía informando de un accidente (que fue notificado por el vehículo involucrado).

- Aplicaciones para la Notificación de Control de Riesgos: Este tipo de aplicaciones está destinada a la notificación entre vehículos de situaciones de posible riesgo en carretera, por ejemplo, curvas muy cerradas, cuestas abajo repentinas, deslizamientos en carretera, entre otras.

- Aplicaciones para la Advertencia de Accidentes Cooperativas: Aplicaciones destinadas a los conductores de vehículos que informan de un posible accidente pudiendo reaccionar a tiempo para cambiar su ruta. Esta es una de las aplicaciones más fuertes en el ámbito del desarrollo de aplicaciones permitiendo la reducción sustancial de los accidentes en carretera.

- Aplicaciones para la Vigilancia del Tráfico**:** Aplicaciones para la vigilancia del tráfico como la instalación de cámaras en las RSU que pueden transmitir información en tiempo real para proporcionar una visión general de la vía.

Las aplicaciones de optimización del tráfico están enfocadas a mejorar el flujo de tráfico, la coordinación de éste y la asistencia en carretera. De igual modo pretende proveer información local actualizada, mapas y mensajes relevantes en tiempo y espacio. Gestión de la velocidad

y navegación cooperativa son dos de los grupos en los que típicamente se clasifican este tipo de sistemas (GONZÁLEZ & CEREZO, 2012).

- Gestión de la velocidad: Tienen por objetivo asistir al conductor para controlar la velocidad de su vehículo de cara a lograr una conducción suave y para evitar paradas innecesarias. Un regulador de velocidad o avisos sobre la velocidad óptima mediante una luz verde son dos ejemplos.
- Navegación cooperativa: Son usados para incrementar la eficiencia del tráfico mediante la cooperación de los navegadores de los vehículos entre ellos o entre éstos y los RSUs. Algunos ejemplos de este tipo son: información del tráfico, itinerario provisional recomendado, control de rutas adaptadas en cooperación y sectorización.

Un gran bloque de aplicaciones puede reseñarse bajo los términos de entretenimiento e información. Aquí la atención se centra en la prestación de servicios a los clientes, automatización de tareas del vehículo o solicitudes de pago, así como en la descarga de música, gestión de flota para empresas dedicadas al transporte, un mantenimiento más sencillo de vehículos, o realizaciones de pago por estacionamientos o por el peaje de las carreteras. La mayoría de estas aplicaciones se centran en el aumento del disfrute y del confort para los usuarios del vehiculares (Campos & Reina, 2014).

- Calculo óptimo de rutas con datos de tráfico en tiempo real: Este servicio puede ser usado tanto desde el propio vehículo como desde cualquier punto conectado a Internet. Podría ofrecerse como un servicio Web que permanentemente informa del estado de las carreteras en tiempo real. El hecho de que a largo plazo todos los vehículos puedan disponer de este sistema facilitaría la identificación y recuento por parte de los equipos instalados en la infraestructura vial a tal efecto. Estos datos convenientemente recogidos y analizados servirán para mostrar el estado y prever futuros atascos.
- Acceso a internet desde los vehículos: Este servicio facilitará el acceso a Internet desde pantallas táctiles dentro de los vehículos. Enmarcado dentro del grupo de ocio y entretenimiento este servicio genérico suplirá posibles carencias en los contenidos del resto de los servicios. Los usuarios podrán acceder a toda la red e

informarse de las condiciones meteorológicas del destino, reservar un hotel e incluso descargar contenidos.

- Información y alertas de gasolineras: A través de este servicio, los usuarios podrán interrogar al sistema de las distancias a las gasolineras más próximas, la empresa y las tarifas de los distintos carburantes. Para ello las diferentes gasolineras que lo deseen entregarán esta información y se comprometerán a tenerla actualizada en cada momento. Una extensión de este servicio, a largo plazo será la integración de sensores en el equipo que informan de la cantidad de combustible restante y con la ruta programada informa al conductor la gasolinera recomendada para repostar.
- Envió de publicidad: Siempre contando con el permiso de los usuarios, se puede desplegar un servicio mediante el cual los equipos de la infraestructura envíen publicidad, básicamente relacionada con los servicios de la vía. Los usuarios podrían configurar sus equipos para aceptar o rechazar este tipo de publicidad incluso definir un perfil con sus preferencias al respecto. Este perfil sería modificable dinámicamente de forma que al circular por autopistas aceptara avisos de publicidad de restaurantes y gasolineras y, circulando por tramos urbanos, anuncios de parkings.

2.2. Herramientas de simulación

Se introduce en profundidad los conceptos referentes a herramientas de simulación, modelos, tipos de simuladores y entre otros para interpretar cuales son los más indicados para el proyecto.

2.2.1. Definición

La simulación en redes VANETs constituye una herramienta valiosa para analizar y evaluar la viabilidad, los beneficios y las bondades de la implementación de las aplicaciones en los sistemas de transporte inteligentes. El grado de realismo y confiabilidad de los resultados de la simulación dependen fundamentalmente de dos aspectos: la integración de un simulador

de red con un simulador de movilidad como en Figura 4, y el uso de métricas adecuadas para la evaluación de los resultados.

El objetivo de integrar un simulador de red y uno de movilidad es crear un escenario realista, en el cual estén consideradas las condiciones de la transmisión de datos a través del canal radio y la interacción de los vehículos bajo un modelo de tráfico real (Ana María Orozco, 2012).

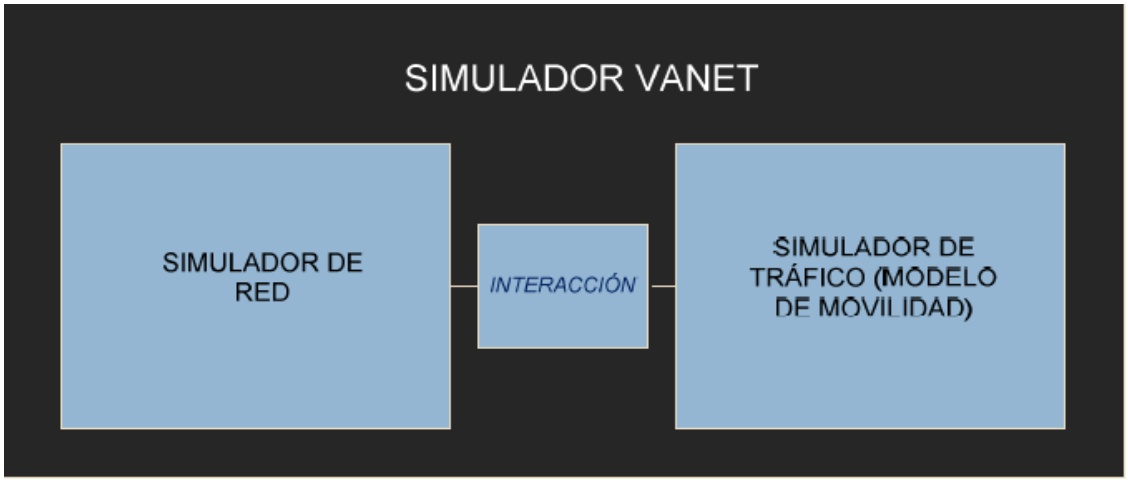

Figura 4. Estructura básica de un sistema de simulación para VANETs
Fuente: (Ana María Orozco, 2012)

2.2.2. Modelos de movilidad

Específicamente, los modelos de movilidad hacen referencia al patrón de movimiento de los nodos en una red ad hoc y determinan la ubicación de éstos en la topología en un instante de simulación dado. Además, describen el cambio de posición, velocidad y aceleración de los nodos en el tiempo. Los modelos de movilidad en redes vehiculares se clasifican de acuerdo a las características con que describen el patrón de movimiento de los nodos. Hay cuatro clasificaciones para los modelos de movilidad: *Modelos Aleatorios*, *Modelos de Tráfico*, *Modelos de Comportamiento* y *Modelos de Flujo,* en la Figura 5 se muestra un resumen de los modelos de movilidad. De estas clasificaciones se destacan los modelos de flujo, los cuales consideran numerosas características de las redes vehiculares y clasifican el nivel de detalle de las variables a tener en cuenta. Por ende, en esta clasificación recae la taxonomía más mencionada en la literatura: *Modelos de movilidad macroscópicos, microscópicos y Mesoscópicos*.

Estos modelos se diferencian por su nivel de detalle al momento de caracterizar redes vehiculares de la siguiente manera: los modelos macroscópicos se basan en las características externas al nodo (tamaño de calles, obstáculos, intersecciones, cruces, paradas, semáforos, entre otras.), además de analizar la densidad y el flujo de tráfico como un conjunto. Los modelos microscópicos se centran en lo que concierne al vehículo en sí (velocidad individual, aceleración, criterios de adelanto, estilo particular de conducción, entre otras.) y los modelos mesoscópicos buscan un equilibrio entre características macroscópicas y microscópicas (O. A. Orozco & Calderón, 2014).

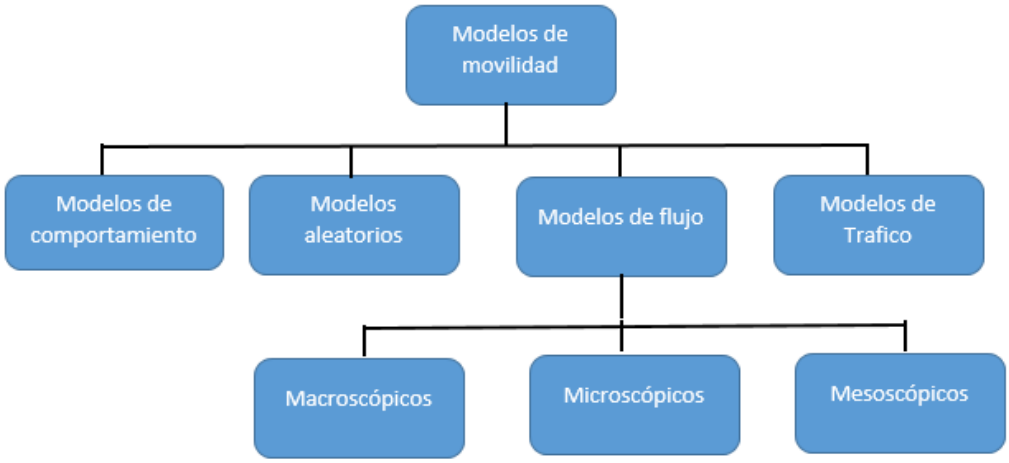

Figura 5. Clasificación de los modelos de movilidad.
Fuente: Adaptado de (Orozco, Chavarro &Calderón, 2013)

2.3.Ataques cibernéticos

En la siguiente sección se describen los diferentes ataques cibernéticos que afectan a las redes VANETs (Sakiz & Sen, 2018).

- Ataque de Posición Falsa: La difusión de información de posición falsa es un problema crítico en VANETs, ya que las aplicaciones relacionadas con la seguridad dependen en gran medida de la información de posición confiable. Además, el análisis de los efectos de la información de posición falsa en VANETs muestra que podría disminuir la relación de entrega de paquetes en general hasta aproximadamente el 90%. Como resultado, la difusión de información de posición falsa podría causar problemas de rendimiento, confiabilidad y seguridad en VANETs.
- Ataque de repetición: En VANETs, los mensajes se pueden almacenar para

reutilizarlos más tarde con el fin de engañar a otras entidades en la red, como en MANET. Esto se conoce como ataque de repetición, y el objetivo es explotar las condiciones en el momento en que se envía el mensaje original. Después de recopilar información que se mueve alrededor de la red, el atacante podría almacenar esa información y reenviarla a la red más adelante, aunque ya no sea verdadera o válida. Además, el ataque podría ser realizado por el remitente original. Por ejemplo, un atacante podría guardar un mensaje recibido sobre un accidente o un evento de tráfico que ocurrió en el pasado y luego reenviarlo. Hasta que el mensaje caduque, el atacante podría reutilizarlo fácilmente para engañar a otros. Sin embargo, la utilización de mecanismos que aseguran la integridad de las marcas de tiempo de los mensajes restringe su probabilidad.

- GPS *spoofing*: Este ataque también se conoce como un ataque de túnel. Un atacante podría inyectar información de posición falsa a otro vehículo(s) utilizando simuladores de GPS. La víctima podría estar esperando una señal de GPS después de abandonar un túnel físico o un área atascada. El simulador de GPS podría generar señales que son más fuertes que las señales de GPS originales. Por lo tanto, incluso un vehículo que recibe una señal original del satélite preferirá aceptar información de posición falsa enviada por el atacante.
- Manipulación del sensor: Dado que la OBU de un vehículo probablemente se instalará en una posición con acceso limitado, un atacante podría tratar de engañar a los sensores simulando condiciones falsas para proporcionar las salidas esperadas. Esa técnica es efectiva porque tal engaño probablemente pasará desapercibido por un sistema de detección dentro del vehículo. Por ejemplo, al frenar en períodos cortos, un atacante podría manipular las aplicaciones relacionadas con la seguridad de tal manera que parezca que hay tráfico en la carretera. Por lo tanto, los mensajes de atasco de tráfico se transmitirán a través de la red.
- Ataque de agujero negro: Mientras que ataques de negación de servicios pretenden deshabilitar la red, otro ataque que da forma a la red es el ataque *Blackhole*. Se conoce como una amenaza seria para los MANET y se refiere a un atacante que manipula a otros nodos para que envíen sus paquetes a través de sí

mismos tanto como sea posible. En VANETs, un vehículo atacante podría explotar protocolos de enrutamiento, como afirmar que tiene el mejor camino para el vehículo de destino / RSU o que está en la mejor posición para reenviar los paquetes. Al transmitir información de enrutamiento falsa, hace que otros vehículos prefieran enviar sus paquetes a través de sí mismos, asumiendo que se encuentra en el camino verdadero. Después de que los vehículos de la víctima mal encaminados envíen sus paquetes al atacante, generalmente solo descarta todos los paquetes de manera intencional y, como resultado, se producen pérdidas de paquetes en la red.

- Ataque de agujero de gusano: Un ataque de agujero de gusano, generalmente se realiza mediante dos o más nodos comprometidos que se involucran en tantas rutas como sea posible al anunciar que conocen el camino más corto hacia cualquier destino. El objetivo del atacante es modificar la topología lógica de la red para recopilar y / o manipular grandes cantidades de tráfico de red. Para realizar el ataque en VANETs, después de recibir un paquete que debe reenviarse, un vehículo atacante encapsula el paquete y lo envía a otro vehículo comprometido. Este último abre el paquete encapsulado y lo propaga. Dado que el paquete original se encapsula durante la transmisión, el campo de conteo de saltos no se puede aumentar, sin importar cuántos saltos haya entre ellos. Por lo tanto, al igual que en el ataque *Blackhole*, esos dos vehículos maliciosos hacen que los protocolos de enrutamiento prefieran el enlace entre ellos como la mejor ruta a cualquier destino, en lugar de rutas más cercanas que ya existen en la red.
- Ataque de *Sybil*: Es el proceso en el que un vehículo reclama estar en diferentes posiciones al mismo tiempo con diferentes identidades. Este ataque dañará toda la topología de la red y requerirá más consumo de ancho de banda. En este modelo de ataque, un vehículo transmite múltiples mensajes con identidades diferentes a los otros vehículos. Todos los demás vehículos suponen que hay un gran tráfico de red. Más adelante se ampliará la información de este ataque (Joe & Ramakrishnan, 2016).
- Ataque de *Timing*: Este ataque colapsa todo el proceso de comunicación de la comunicación de la red ad hoc del vehículo. Ya que el objetivo fundamental de

la comunicación VANETs es difundir los mensajes de emergencia a tiempo. Este ataque de tiempo será el mayor desafío para los investigadores. En VANETs, cada mensaje debe transmitirse a tiempo sin más demora para lograr el mejor rendimiento de la comunicación. Más adelante se ampliará la información de este ataque (Joe & Ramakrishnan, 2016).

2.4. Mecanismos de verificación de plausibilidad

Existen diferentes enfoques para la detección del mal comportamiento que se pueden usar para categorizar los diferentes mecanismos de detección. En primer lugar, el enfoque centrado en los nodos que requiere de mecanismos de autenticación para distinguir la confiabilidad de la comunicación entre los nodos. Muchos sistemas logran esto asumiendo una confianza de terceros como una PKI (*Public Key Infrastructure*) que emite credenciales, que luego se utilizan para autenticar mensajes y la correspondiente información utilizando un mecanismo de seguridad como firmas digitales. Los mecanismos centrados en los nodos se dividen en mecanismos conductuales y basados en la confianza. Por otro lado, el enfoque centrado en los datos abarca todos los mecanismos que inspeccionan directamente la información difundida para detectar posibles comportamientos indebidos. Estos mecanismos centrados en datos a menudo todavía requieren de alguna forma de vinculación entre mensajes para ser capaz de distinguir confiablemente entre diferentes nodos. Sin embargo, estos mecanismos no dependen de la vinculación de mensajes, lo que los hace muy valiosos para la detección de ataques Sybil. En respuesta a esto, muchos investigadores de VANETs han desarrollado nuevos esquemas para la detección de mal comportamiento centrada en datos, dividiéndolos en mecanismos de consistencia y plausibilidad.

Estos últimos mecanismos de plausibilidad son aquellos que tienen algún modelo implícito o explícito del mundo real, y verifican sí la información entrante es aceptable dentro de este modelo. Por ejemplo, en VANETs los informes de velocidad de 700 km/h no son muy aceptables y deberían ser filtrados de los datos a analizar. Sin embargo, la plausibilidad debe aplicarse con precaución en VANETs, como parte del enfoque de tales redes es detectar valores atípicos que indican algún grado de importancia, como congestiones y colisiones entre vehículos (Heijden, Dietzel, & Kargl, 2014).

2.5.Estado del arte

Título	Autores	Año	Resumen
VeReMi: A Dataset for Comparable Evaluation of Misbehavior Detection in VANETs	Rens W. van der Heijden, Thomas Lukaseder, and Frank Kargl	2018	Este proyecto tenía como objetivo dar a conocer que los estudios comparativos entre los mecanismos de detección son raros debido a la falta de una referencia de conjuntos de datos, lo que los llevo a tomar enfrentar este desafío introduciendo el conjunto de datos de mal comportamiento de referencia vehicular (VeReMi) y una discusión de métricas válidas para dicha evaluación. VeReMi es el primer público extensible conjunto de datos, permitiendo a cualquiera reproducir el proceso de generación, así como contribuir con ataques y usar los datos para comparar nuevas detecciones. Mecanismos contra los existentes. (Heijden, Lukaseder, & Kargl, 2018)
A survey of attacks and detection mechanisms on intelligent transportation systems: VANETs and IoV	atih Sakiz , Sevil Sen	2018	Este documento tiene como objetivo examinar los posibles ataques y los mecanismos de detección correspondientes que se proponen en la literatura. Los ataques se clasifican y explican junto con sus efectos, y las soluciones se presentan junto con sus ventajas y desventajas. (Sakiz & Sen, 2018)
Feature Selection for Anomaly Detection in Vehicular Ad Hoc Networks	Van Huynh Le, Jerry den Hartog y Nicola Zannone	2018	Este documento se basó en la investigación de aplicaciones de detección de anomalías de la caja blanca para detectar dichos ataques. Un paso clave para aplicar este enfoque es la selección de las características de comportamiento "correctas", es decir, las características que permiten la detección de ataques y proporcionan una comprensión de las alertas planteadas. Al encontrar características significativas y construir modelos precisos de comportamiento normal, este trabajo es un primer paso hacia el diseño de motores de detección de anomalías eficaces para la comunicación V2V. (Le, Hartog, & Zannone, 2018)
Classification of Security Attacks in VANETs: A Review of Requirements and Perspectives	Mohammed Ali Hezam Al Junaid, Syed A. A, Mohd Nazri Mohd Warip, Ku Nurul Fazira Ku Azir y Nurul Hidayah Romli	2018	En este documento, se evaluaron los problemas de seguridad de VANETs y se discutieron los desafíos en VANETs. Igualmente importante, revisión comparativa de los requisitos de seguridad, el tipo de ataques y las capacidades de los atacantes presentes en VANETs. (Junaid, A, Warip, Azir, & Romli, 2018)
Performance Evaluation and Detection of Sybil Attacks in Vehicular Ad-Hoc Networks	Jyoti Grover, Deepak Kumar, M. Sargurunathan, M.S. Gaur, y Vijay Laxmi	2010	Este documento se basó en la detección del ataque *Sybil*, presentando una implementación del escenario de ataque simulado en VANETs y su discusión de impacto del rendimiento de la red. También se presenta un enfoque cooperativo de detección de ataques de *Sybil*, inferido a través

			del análisis del ataque de *Sybil*. (Grover, Kumar, Sargurunathan, Gaur, & Laxmi, 2010)
Detecting Sybil attacks in VANETs	Bo Yu, Cheng-Zhong Xu, Bin Xiao	2013	En este documento, se realizaron varios intentos para explorar la posibilidad de detectar ataques de *Sybil* mediante el análisis de la distribución de la intensidad de la señal. Primero, se propone un método cooperativo para verificar las posiciones de los posibles nodos de *Sybil*. Utilizando un algoritmo basado en RANSAC (*Random Sample Consensus*) para hacer que este método cooperativo sea más sólido contra los datos atípicos fabricados por los nodos de *Sybil*. Introduciendo un método estadístico y diseño de un sistema que puede verificar de dónde proviene un vehículo Finalmente, se realizaron simulaciones para evaluar la viabilidad y eficiencia de sus métodos. Su esquema demuestra ser un enfoque económico para suprimir los ataques de *Sybil* sin soporte adicional de hardware de posicionamiento específico. (Yua, Xu, & Xiao, 2013)
Redes Vehiculares Aplicadas a la Movilidad Inteligente y Sostenibilidad Ambiental en Entornos de Ciudades Inteligentes	José Antonio Sánchez Sánchez	2017	Se basa en el término de Ciudad Inteligente (*Smart City*). Este concepto se fundamenta en la propuesta de soluciones innovadoras que doten a una ciudad de servicios destinados a facilitar la vida de sus habitantes y que favorezcan el desarrollo de ciudad como conjunto. En este ámbito, los ITS, han supuesto un relevante auge en el desarrollo de nuevas propuestas tecnológicas aplicadas al mundo del transporte. Dentro de los ITS, surgió el concepto de VANETs. (Sánchez, 2017)
An evaluation of reputation with regard to the opportunistic forwarding of messages in VANETs	Luz M. Santos J. y Edson Moreira	2019	Este trabajo propone llevar el registro del comportamiento de los vehículos en VANETs mediante un sistema de reputación centralizado. Se realizan una serie de simulaciones en un escenario urbano para obtener las reputaciones de encaminamiento y generación de mensajes para cada uno de los vehículos (J. & Moreira, 2019).
Malicious Node Detection in Vehicular Ad-Hoc Network Using Machine Learning and Deep Learning	Elvin Eziama, Kemal Tepe, Ali Balador, Kenneth Sorle Nwizege y Luz M. S. Jaimes	2018	Este trabajo presenta un modelo de confianza con respecto al aprendizaje automático/profundo (ML/DL), proporciona un enfoque basado en datos para resolver los desafíos de seguridad en redes dinámicas como las VANETs. Modela la confianza como un proceso de clasificación y extracción de características relevantes usando un modelo híbrido como *Bayesian* Neural *Network* que combina el aprendizaje profundo con modelado probabilístico para una decisión inteligente y una generalización efectiva en la confianza. (Eziama, Tepe, Balador, Nwizege, & Jaimes, 2018)

Tabla 2. Relación del estado del arte
Fuente: Propia

3. DESARROLLO DE SIMULACIONES

El presente capítulo tiene como fin introducir en detalle conceptos, variables, recursos y procedimientos utilizados en el estudio e implementación de las simulaciones del proyecto. Tomando como punto de partida el proyecto VeReMi, donde se pretende extender la información ya establecida agregando dos tipos de ataques cibernéticos simulados (*Attack Sybil y Attack Timing*) que posteriormente serán explicados en profundidad.

No obstante, se utilizaron las mismas herramientas de simulación de código abierto en las que se basó el proyecto VeReMi. En la Tabla 3 se describe las versiones de los simuladores y los recursos usados en el desarrollo del proyecto.

Simuladores y recursos	Versiones
Simulador de movilidad	Sumo: 0.30.0
Simulador de red	OMNeT++: 5.3
Simulador hibrido	VEINS: 4.6
Lust	2.0
Equipo Portátil	ASUS – X542U
RAM	12 GB
Procesador	Core i7
Sistema operativo	Windows 10 Pro

Tabla 3. Simuladores y recursos
Fuente: Propia

3.1. Estudio de VEINS

Primero que todo VEINS es un Framework de simulación para IVC (*Comunicación Inter-Vehicular*) de código abierto, donde se envía un conjunto de modelos de simulación de redes vehiculares. Estos modelos son ejecutados por un simulador de red basado en eventos mientras interactúan con un simulador de tráfico. Estos simuladores son: OMNeT++ para la simulación de la red y SUMO para la simulación de tráfico. Para llevar a cabo evaluaciones de IVC, ambos simuladores corren en paralelo, conectados a través de un socket TCP. El protocolo para esta comunicación se ha estandarizado como la Interfaz de Control de Tráfico TRACI (*Traffic Control Interface*). Esto permite la simulación de acoplamiento bidireccional del tráfico por carretera y el tráfico de red, donde el movimiento de vehículos en el simulador

de tráfico de carretera SUMO se refleja en el movimiento de los nodos en una simulación OMNeT++ (Sommer, 2006).

En segundo lugar, VEINS en colaboración con OMNeT se comunica por eventos discretos en la red mediante el intercambio de mensajes entre sus módulos, y maneja una codificación para la creación de sus aplicaciones y archivos en lenguaje NED (OMNeT), orientada a objetos y desarrollado en C++. Estos mensajes tienen como formato las siguientes variables: tiempo de recepción, el tiempo de transmisión reclamado, el remitente reclamado, un ID de mensaje único de simulación, un vector de posición, un vector de velocidad, el RSSI. En la figura 6 se muestra el formato de los mensajes del VEINS original con las variables anteriormente descritas, posteriormente en las Tablas 4 y 5 se encuentran los archivos de configuración utilizados en la implementación del proyecto y algunos ejemplos de parámetros y codificación de estos archivos.


```
{
"type":3,
"rcvTime":1000.437698600892,
"sendTime":1000.437587209977,
"sender":7,
"messageID":104,
"pos":[3602.4173636345715,5518.285008013137,1.895],
"pos_noise":[0.0,0.0,0.0],
"spd":[-2.447519111077679,29.877753957007525,0.0],
"spd_noise":[0.0,0.0,0.0],
"RSSI":4.478375873553147e-8
}
```


Figura 6. Formato de VEINS original
Fuente: Propia basado en (Heijden, Lukaseder, & Kargl, 2018)

Archivo de configuración	Ubicación	Nota
Omnetpp.ini	~\simulations\lust\omnetpp.ini	En ellos se describen los parámetros característicos de la simulación, el número de simulaciones y repeticiones a ejecutar, la preparación y duración de la simulación, el valor de los atributos de los módulos simples, atributos del escenario, aplicaciones a ejecutar, estándares 802.11p y 1609, movilidad y entre otros.
AttackerApp.h	~\src\veins\modules\application\attackerapp\ AttackerApp.h	En este archivo se inicializan las variables a utilizar, las funciones necesarias para la implementación de la aplicación y se definen los nombres e id de los ataques.
AttackerApp.cc	~\src\veins\modules\application\attackerapp\ AttackerApp.cc	En este archivo se llaman las funciones inicializadas en el AttackerApp.h y se implementan. Además en este archivo es donde se actualizan, envían y reciben los mensajes, y encapsulan cada uno de estos en los archivos json.
AttackerApp.ned	~\src\veins\modules\application\attackerapp\ AttackerApp.ned	Este archivo contiene las variables que necesita la aplicación que son inicializadas en el omnetpp.ini y es encargado de la comunicación entre estos.
TraCIMobility.h	~\src\veins\modules\mobility\traci\TraCIMobility.h	En este archivo se inicializan variables y funciones necesarias para movilidad del modelo TRACI. Y también es donde implementan funciones que permiten llamar variable de otros archivos de movilidad.
BaseMobility	~\src\veins\base\modules\BaseMobility	Este archivo proporciona una ubicación aleatoria de los host y pantalla, actualizaciones, notificaciones de cambio sobre los cambios de posición y manejo de frontera en la movilidad.
BaseWaveApplLayer.h	~\src\veins\modules\application\ieee80211p\BaseWaveApplLayer.h	Este archivo permite crear los eventos de atención a los mensajes que se van a utilizar en la aplicación.
WaveShortMessage_m	~\src\veins\modules\messages\WaveShortMessage	Los archivos de este tipo permiten crear funciones que devuelvan una variable determinada, pero solo para mensajes BSMs o WSAs.

Tabla 4. Archivos de configuración estudiados
Fuente: Propia

Archivos de configuración	Ejemplos
Omnetpp.ini	• ***.node[*].appl.attackerType = 5** #especifica el tipo de ataque agregando el número correspondiente al ataque a simular. • **repeat = 5** # especifica las repeticiones a realizar en la simulación. • ***.manager.firstStepAt = ${start = 3,5,7}h** # especifica el tiempo de la preparación de la simulación. Se le puede ajustar una densidad mayor o menor dependiendo de la cantidad de datos necesarios de salida. • ***.node[*].appl.attackerProbability = ${0.1,0.2,0.3}**# describen la probabilidad de aparición de los nodos atacantes. Se puede modificar las probabilidades dependiendo de la cantidad de atacantes que necesite el investigador a la hora de simular • **sim-time-limit = ${start}h+220s** #tiempo de simulación. Se modifica el valor en segundos a gusto del investigador, dependiendo del tiempo colocado y de la preparación de la simulación pueden dar como salida una cantidad baja o amplia de datos. • **.manager.configFile = "dua.static.sumocfg"** #llama al archivo de conexión con sumo. • ***.manager.launchConfig = xmldoc("luxembourg.launchd.xml")** # llama el archivo donde se describe el mapa de la simulación. • ***.node[*].applType = "AttackerApp"** #tipo de aplicación a ejecutar. • ***.node[*].appl.beaconInterval = 1s** #intervalo de tiempo en él envió de mensajes BSMs. • ***.node[*].appl.traceGroundTruthJSONFile = "GroundTruthJSONlog"** #describe la ubicación donde se guardan los mensajes de verdad de toda la simulación. • ***.node[*].appl.traceJSONFile = " JSONlog-"** # describe la ubicación donde se guardan los mensajes de la simulación por nodo.

AttackerApp.h	• **#define ATTACKER_TYPE_NO_ATTACKER 0** //se define el tipo del ataque y un identificador para navegar en el proyecto.
	• **int** attackerType; //se crea la variable tipo de ataque.
	• **virtual void onBSM(BasicSafetyMessage* bsm);** // Se crea la función polimórfica para llenar el dataset

AttackerApp.cc	• **void AttackerApp::initialize(int stage)** //En esta función se inicializan las variables creadas en el archivo AttackerApp.h, se extraen los valores del archivo omnetpp.ini y se planifica los eventos a utilizar en la creación de los mensajes • **sendGhost1evt = new cMessage ("ghost1", *SEND_GHOST1_EVT*);** // se inicializa un evento de tipo cMessaage, con el fin de usarlo en la implementación de ataques cibernéticos. • **scheduleAt(simTime() + beaconInterval+ghost1, sendGhost1evt);** //En esta línea se planifica el evento agregándole al intervalo de beacon un tiempo para el envió del mensaje. • **void AttackerApp::onBSM(BasicSafetyMessage* bsm)** //En esta función se reciben los mensajes BSMs de los nodos, se agregan las variables a guardar en el dataset y en donde se crean los archivos json para cada nodo. • **void AttackerApp::populateWSM(WaveShortMessage* wsm, int rcvId, int serial)** // En esta función se reciben los mensajes BSMs de todos los nodos, se agregan las variables a guardar en el dataset y en donde se crea el archivo de json de verdad. • **void AttackerApp::attackBSM(BasicSafetyMessage* bsm, int x)** // En esta función se muestra un menú con todos los tipos de ataque realizado y es el encargado de direccionar a la funciones de cada ataque. • **void AttackerApp::attackSetFANTASMA(BasicSafetyMessage***

	bsm,int x) // En esta función se crea los mensajes fantasma, cambiando el id del módulo, vector de velocidad y de posición. • **bsm->setSenderAddress(FANTASMA+1000); //** Esta línea me permite extraer el modulo del nodo atacante y agregarle un valor para diferenciarlo de otros nodos obteniendo un módulo falso.
AttackerApp.ned	• **double attackerProbability = default(0); //**Me conecta la variable attackerProbability con el archivo omnetpp.ini y los demás archivos del proyecto VEINS.
TraCIMobility.h	• **virtual Coord getDirection() const {** **return move.getDirection();** **}** Esta función me permite extraer la dirección de los nodos al moverse por el mapa en el transcurso de la simulación. Al igual que esta función en este archivo se pueden extraer variedad de variables pertenecientes a la movilidad de los nodos.
BaseMobility	• **virtual void updatePosition(); //** Esta función me permite establecer si la posición del nodo ha cambiado y también me permite mover el icono del Host a la nueva posición en la pantalla.
BaseWaveApplLayer.h	• **enum WaveApplMessageKinds {** ***SEND_BEACON_EVT, SEND_GHOST1_EVT, SEND_GHOST2_EVT, SEND_DENSITY_EVT, SEND_TIMING_EVT, SEND_ACCIDENT_EVT, SEND_WSA_EVT*** **};** **//** Esta función me permite crear el nombre del evento a implementar. • **virtual void sendDown(cMessage* msg); //** Esta función me permite enviar los mensajes de cada nodo a los nodos cercanos.

Tabla 5. Ejemplos de archivos de configuración
Fuente: Propia

3.2. dataset

La primera contribución de este documento es un conjunto de datos destinado a proporcionar una línea de base común para la evaluación de detección de mala conducta. Los estudios anteriores siempre se han basado en estudios de simulación diseñados individualmente, aunque esto tiene la ventaja de ataques personalizables debido a la especificidad del escenario, es difícil comparar los mecanismos entre sí. El propósito del conjunto de datos es proporcionar una línea de base inicial con la que se puedan comparar los mecanismos de detección. Esto reduce el tiempo requerido para que los investigadores realicen estudios de simulación de alta calidad, y facilita a los lectores comparar los resultados de diferentes trabajos (Heijden, Lukaseder, & Kargl, 2018).

El dataset consiste en registrar los mensajes recibidos por cada vehículo y un archivo llamado *GroundTruth* que describe el comportamiento del atacante. La información que se establece

por vehículo se divide en dos tipos de mensajes diferentes: una que contiene datos GPS que son mensajes que se van actualizando en determinado intervalo de tiempo (etiquetados type=2) y los mensajes BSM (etiquetados de type=3) que son aquellos en los que se basa la mayor parte de implementación del proyecto, ya que facilita la interacción con el concepto de aplicaciones de seguridad anteriormente explicadas. El GroundTruth es el que almacena los mensajes legítimos BSMs de todos los vehículos (etiquetados de type=4), lo cual permite a cualquier detector determinar la confiablidad de cada mensaje almacenado. El dataset consta de 90 simulaciones individuales, con dos ataques diferentes (ataque *Sybil* y *Timing*), tres probabilidades diferentes en cada ataque, tres densidades de tráfico diferentes y cinco repeticiones (con diferentes semillas aleatorias).

3.2.1. Estructura de archivos

Cada simulación al finalizar arroja un conjunto de archivos con extensión JSON. El archivo *GroundTruth* con todos los mensajes generados por los vehículos y archivos individuales por vehículo con el conjunto de registros de mensajes recibidos. El nombre de cada archivo se identifica con el número del vehículo, su número de módulo OMNeT ++ y además de su estado, así como se muestra en la Figura 7.

Figura 7. Estructura de archivo
Fuente: Propia

3.3. Escenario

Al igual que el proyecto VeReMi se toma el escenario de tráfico de la ciudad de Luxemburgo (*Lust*), que proporciona un escenario completo para la evaluación de las aplicaciones VANETs. Por otra parte, VeReMi proporciona una explicación de la situación actual de comparación de escenarios de detección en estas redes y a su vez da una posible solución que es proporcionar un flujo de mensajes simple por vehículo, lo que facilita la reproducción de

los estudios de detección. La Tabla 6 describe algunos parámetros centrales de la simulación. Cada uno de estos parámetros se pueden modificar y adaptar al escenario que el investigador o investigadores estén utilizando en la creación de sus simulaciones.

Con relación a creación del dataset se pueden evidenciar muchas similitudes en diferentes proyectos establecidos en otros escenarios, como por ejemplo el escenario de la ciudad de sao pablo, sao carlos, Brasil presentado por (J. & Moreira, 2019).

Parámetros	Valores	Nota
Mobility	SUMO lust (DUA estática)	lust v.2
Simulation start	(3,5,7)h	Controla la densidad
Simulation duration	600s	Duración de simulación
Attacker probability	(0.1,0.2,0.3)	Atacantes con estas probabilidades
Area simulation	2300,5400-6300,6300	Varios tipos de carreteras
Signal interference model	Interferencia	VEINS por defecto
Obstacle shadowing	Simple	VEINS por defecto
Fading	Jakes	VEINS por defecto
Shadowing	Log-Normal	VEINS por defecto
MAC implementation	802,11p	VEINS por defecto
Thermal noise	-110dbm	VEINS por defecto
Transmit Power	20Mw	VEINS por defecto
Bit rate	6 Mbps	VEINS por defecto (mejor recepción)
Sensitivity	-89dBm	VEINS por defecto
Antenna Model	Monopole en raíz	VEINS por defecto
Beaconing	1Hz	VEINS por defecto
Channel	CCH	VEINS por defecto (falso)
BeaconInterval	1 s	Periodo definido para retransmisión de los mensajes.

Tabla 6. Parámetros de simulación
Fuente: (Heijden, Lukaseder, & Kargl, 2018)

3.4. Ataques e implementación

En esta sección se describe en detalle los ataques a simular *Sybil y Timing*, su forma de operar y vulnerar a las redes VANETs. También se presenta la implementación de cada uno de ellos en el código abierto que se dispuso a extender del proyecto VeReMi.

3.4.1. Ataque Sybil

Como se explicó anteriormente en la sección 2.3 este ataque se puede clasificar como uno de los ataques más peligrosos en VANETs. En un escenario de ataque de *Sybil*, un nodo (vehículo) puede simular que tiene más de una identidad. En otras palabras, otros nodos en la red no pueden distinguir si la información proviene de un vehículo o de más de un vehículo. El objetivo principal del atacante es dar forma a las redes en función de sus objetivos. Por ejemplo, un atacante podría manipular los comportamientos de otros vehículos, como hacer que tomen un camino diferente de su ruta programada.

Además de ser una de las formas de ataque más peligrosas, este ataque también se encuentra entre los más difíciles de detectar. Se vuelve más arriesgado en redes que usan enrutamiento geográfico, ya que el atacante afirma que el vehículo está en varias posiciones al enviar información incorrecta sobre su posición. Además, podría mostrar eventos que ocurren en posiciones distintas a sus posiciones genuinas (Sakiz & Sen, 2018) y (Yua, Xu, & Xiao, 2013).

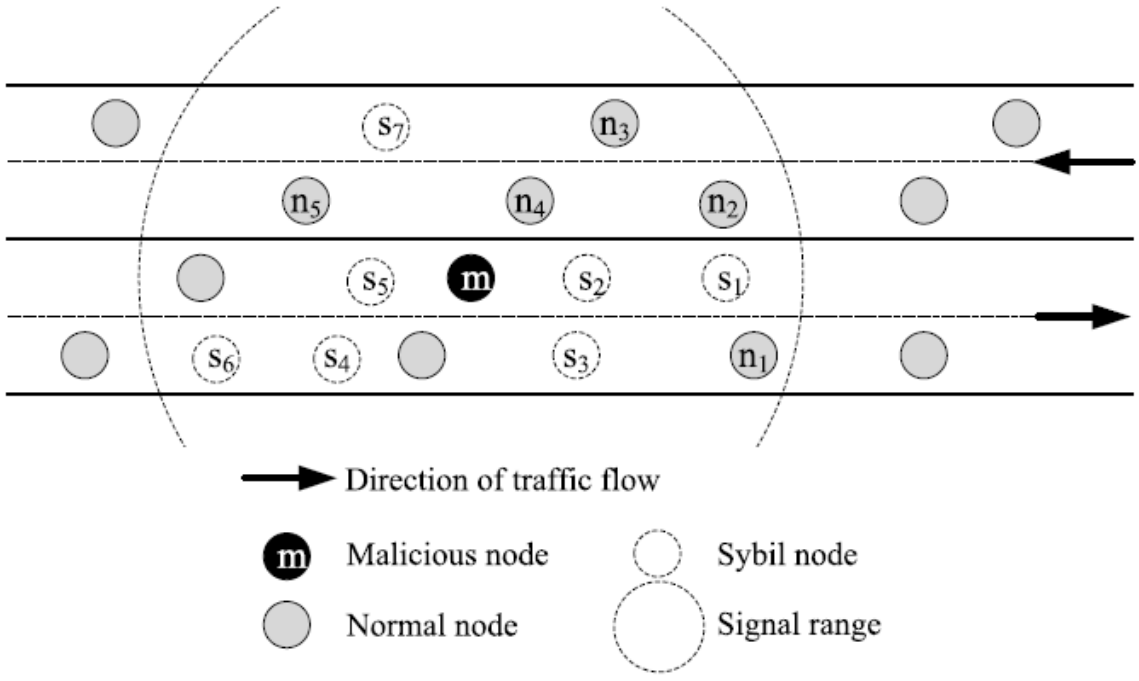

Figura 8. Ejemplo de ataque Sybil
Fuente: (Yua, Xu, & Xiao, 2013)

En la Figura 8 se puede observar un ejemplo de este ataque. En una carretera se presentan tres tipos de nodos: los grises que son nodos legítimos que van recorriendo la vía normalmente, un segundo nodo de color negro que es el malicioso el cual se encargado de crear los nodos fantasmas (nodos *Sybil*) en el rango de señal DSRC. Por lo tanto, se puede decir que hay una comunicación entre el nodo n1 y nodo s1, pero la recepción de los mensajes va a ser diferente. Porque el mensaje recibido por n1 va tener el id y módulo de s1 en cambio los mensajes que llegan a s1 serán reenviados al nodo M que es el malicioso y creador de la identidad de s1.

3.4.2. Ataque Timing

Como se conceptualizo previamente en la sección 2.3, este ataque es peligroso por la congestión y accidentes que puede ocasionar en la comunicación de la red de las VANETs. No obstante, hay dos tipos de ataque *Timing* (SUMRA, MANAN, & HASBULLAH, 2014): El ataque *Timing* básico, tiene una comunicación de tipo *unicast* entre un remitente y un receptor. Ya que su comunicación solo necesita de dos nodos es un ataque muy poco frecuente y no es tan peligroso a la hora de generar congestión. La Figura 9 explica la situación en la que el vehículo B es un vehículo atacante y lanza un ataque de tiempo entre el vehículo A y el vehículo C. Sí el vehículo C recibe este mensaje en el momento adecuado, C tendrá dos opciones para moverse, ya sea en la dirección A o B. En un caso grave, si el vehículo C recibe el mensaje correcto que se retrasa debido al ataque de tiempo, el vehículo C se dirigirá a una carretera equivocada en la que realmente se produjo el accidente. La Figura 10 muestra la ubicación del vehículo C, en donde gira hacia la carretera incorrecta debido a un retraso en el mensaje.

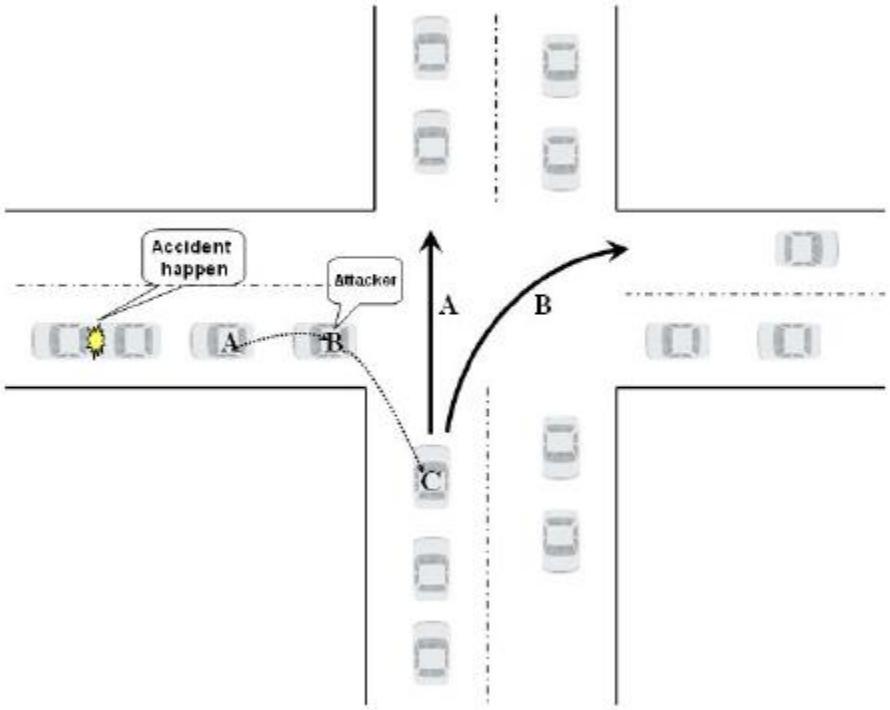

Figura 9. Ataque Timing básico (antes)
Fuente: (SUMRA, MANAN, & HASBULLAH, 2014)

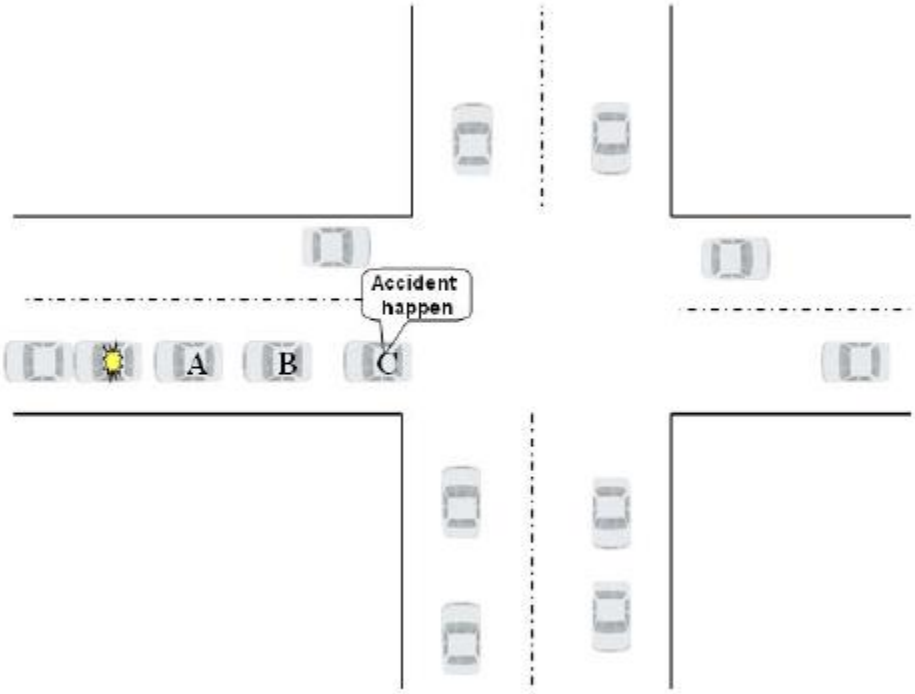

Figura 10. Ataque Timing básico (después)
Fuente: (SUMRA, MANAN, & HASBULLAH, 2014)

El ataque *Timing* extendido a diferencia del anterior es de tipo *Broadcast*, en este tipo de ataque un nodo malicioso vulnera a un grupo de nodos legítimos causando que todos tomen la decisión incorrecta por la llegada tarde del mensaje, ocasionando así una congestión vehicular.

La Figura 11 muestra cómo el atacante lanza un ataque de tiempo usando una técnica de multidifusión en la que el atacante ataca a un grupo específico de vehículos. En este caso, múltiples usuarios se ven seriamente afectados.

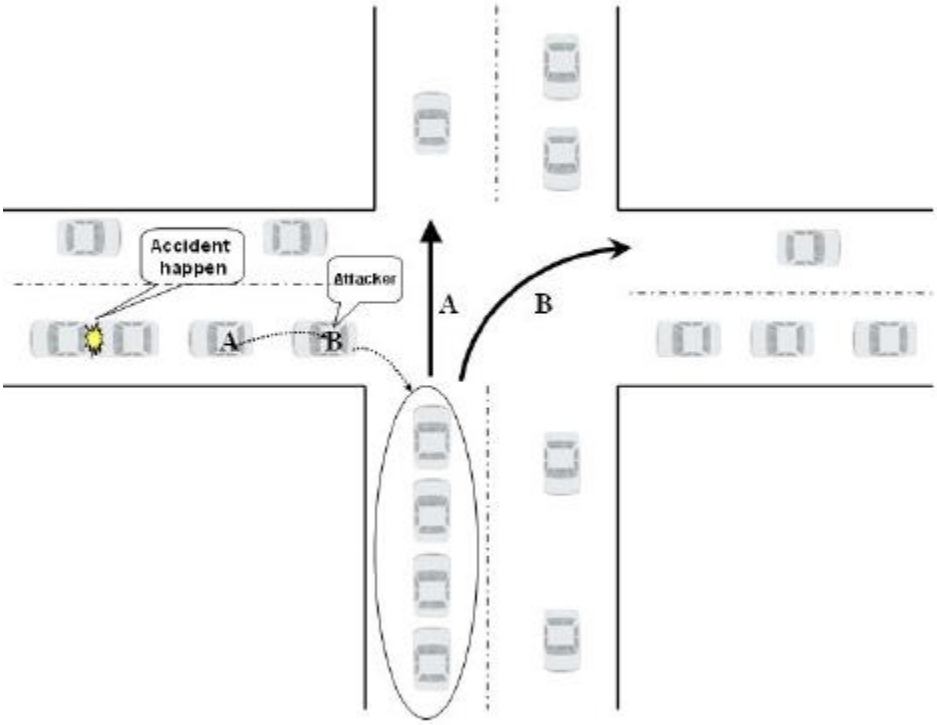

Figura 11. Ataque Timing básico (antes)
Fuente: (SUMRA, MANAN, & HASBULLAH, 2014)

3.4.3. Implementación de ataques

En esta parte se amplían las implementaciones del proyecto VeReMi asociadas a falsa posición, con dos nuevos ataques cibernéticos. En primer lugar, el ataque *Sybil* que es uno de los más peligrosos en las redes VANETs por su falsificación de identidad y posición al crear nuevos nodos fantasma que vulneran a un objetivo u objetivos haciendo creer de la

existencia legítima de estos, por medio de la comunicación de los mensajes. Por otro lado, se implementa de igual manera el ataque *Timing*, como se explicó anteriormente es un ataque que colapsa la comunicación de la red retrasando el envío de un mensaje de emergencia ocasionando que este llegue tarde a los nodos que están cerca del rango de señal DSRC, provocando congestión vehicular. A su vez en este proyecto se implementó el ataque *Timing* extendido por su tipo de difundir los mensajes y su impacto en las redes VANETs.

Al implementar los ataques en el código abierto dejado por VeReMi se estableció la siguiente dinámica para cada ataque: En el ataque *Sybil* se crearon dos tipos de eventos, cada uno para simular un nodo fantasma donde estos se enviaban en tiempos menores que el *beaconInterval* (1s). También cada nodo fantasma se le modificaba su posición, velocidad y módulo OMNeT++ al enviar los mensajes desde el nodo malicioso a los demás nodos legítimos en un rango de señal DSRC. Para modificar la posición de los dos fantasmas se buscó que uno quedara adelante del nodo malicioso y el otro atrás, utilizando dos valores en eje el X y Y para agregar y disminuir la posición original.

Por otra parte, en el ataque *Timing* se crea un evento que facilita realizar el accidente en un nodo especifico (nodo accidente), este nodo envía el tiempo del accidente y el modulo del nodo que será el atacante. Para hallar este módulo se crea una lista con todos los nodos cercanos al nodo accidente en el rango de señal DSRC y se retorna el módulo del primer nodo de la lista. El nodo seleccionado como atacante se auto habilita al reconocer su id en el mensaje que reporta el accidente, y este es el encargado de retardar el mensaje programando su envío un intervalo de tiempo más al tiempo actual, de tal manera que al enviar los mensajes a los nodos legítimos estos reciben el mensaje de tiempo de accidente demasiado tarde. Los parámetros para los ataques se muestran en la Tabla 7.

ID	Ataques	Parámetros
3	Sybil	X = 300 y Y= -250
5	Timing	AccidentCount = 1

Tabla 7. Parámetros de ataques
Fuente: Propia

3.5. Características

El conjunto de datos consta de un total de 90 simulaciones, divididas en tres tipos de densidades. La baja densidad (tráfico que comienza a las 3:00) tiene de 180 a 192 vehículos, mientras que la densidad media (tráfico que comienza a las 5:00) tiene entre 460 y 475 vehículos, y la alta densidad (tráfico que comienza a las 7:00) tiene entre 816 y 843 vehículos. Una parte de estos vehículos considerados los maliciosos, se toman muestreando una distribución uniforme [0; 1] y comparándola con el parámetro de fracción de atacante, esencialmente asignando a cada vehículo un atacante con esa probabilidad (Heijden, Lukaseder, & Kargl, 2018). Todos los vehículos clasificados como atacantes ejecutan el tipo de ataque que le corresponde (explicados en las secciones anteriores). Cada receptor contiene registrados las siguientes variables, dependiendo del ataque a simular. En el ataque *Sybil* son: tipo de mensaje, tiempo de recepción, tiempo de transmisión, remitente, id del mensaje, ángulo, velocidad, aceleración, densidad (esta variable hace referencia a la cantidad de nodos que se encuentran alrededor de cada nodo receptor), distancia, CO2, radio de giro, *yaw rate*, vector de dirección, vector de posición, vector de velocidad y el RSSI. Para el ataque *Timing* son las mismas variables a excepción de dos, que son tiempo de accidente y modulo accidente. Además, un archivo de verdad que se actualiza cada vez que un vehículo envía un mensaje este archivo contiene las siguientes variables dependiendo del ataque a simular. En el ataque *Sybil* son: tipo de mensaje, tiempo de transmisión, tipo de ataque, remitente, id del mensaje, ángulo, velocidad, aceleración, densidad, distancia, CO2, radio de giro, *yaw rate*, vector de dirección, vector de posición, vector de velocidad y el RSSI. Para el ataque *Timing* son las mismas variables a excepción de dos, que son tiempo de accidente y modulo accidente. En la tabla 8 se describen las variables que son adicionadas respecto a las del proyecto VeReMi y los proyectos que las mencionan. Las figuras 12 y 13 me describen la estructura de los formatos json del ataque *sybil* y *Timing*.

Variables	Contribución a estas redes	Proyectos
Velocidad	Esta variable es importante ya que permite determinar algunos cálculos matemáticos en la detección de ataques cibernéticos, y de ayuda en la validación	1. Impact of the speed and mobility model in a data communication for a vehicular network. (O. A. Orozco & Calderón, 2014) 2. Survey on sybil attack defense mechanisms in wireless ad hoc

	e interpretación de investigación sobre las redes VANETs.	networks. (Vasudevaa & Sood, 2018) 3. Privacy-Aware VANETs Security: Putting Data-Centric Misbehavior and Sybil Attack detection Schemes into Practice. (Hussain, Kim, & Oh, 2012) 4. Power aware malicious nodes detection for securing MANETs against packet forwarding misbehavior Attack. (Kukreja, Dhurandher, & Reddy, 2017) 5. Performance Evaluation and Detection of Sybil Attacks in Vehicular Ad-Hoc Networks. (Grover, Kumar, Sargurunathan, Gaur, & Laxmi, 2010)
Ángulo de giro	Esta variable es utilizada para evitar colisiones a la hora de cambiar de ruta y a su vez sirve para calcular otras variables importantes como el radio de giro y el *yaw rate*.	1. Performance Evaluation and Detection of Sybil Attacks in Vehicular Ad-Hoc Networks. (Grover, Kumar, Sargurunathan, Gaur, & Laxmi, 2010) 2. Survey on sybil attack defense mechanisms in wireless ad hoc networks. (Vasudevaa & Sood, 2018)
Aceleración	Este tipo de variable se usa normalmente en los modelos de movilidad, que utilizan el término microscopia para describir el comportamiento del este. A su vez es utilizada en la detección de ataques.	1. VANETs applications focused on environmental sustainability, a systematic review. (Sarasti & Ramírez, 2014) 2. Impact of the speed and mobility model in a data communication for a vehicular network. (O. A. Orozco & Calderón, 2014) 3. Secure Timing Advance Based Context-Aware Handover Protocol for Vehicular Ad-Hoc Heterogeneous Networks. (Nyangaresi, Abeka, & Rodgrigues, 2018)
Densidad	Esta variable aunque no es muy usada, sirve como ayuda en la implementación de ataques a simular.	1. Privacy-Aware VANETs Security: Putting Data-Centric Misbehavior and Sybil Attack Detection Schemes into Practice. (Hussain, Kim, & Oh, 2012)
C02	Esta variable es muy utilizada para determinar la cantidad de este gas en los vehículos y con ello poder buscar soluciones para disminuir la producción de este gas en los vehículos y así evitar dañar a las personas.	1. Soluciones para la autenticación y gestión de subredes en MANETs y vanets (GIL & GIL, 2012) 2. OSA: una aplicación VANETs enfocada en la eficiencia energética. (Orozco & Gonzalo Llano, 2014)
Yaw rate	Esta variable es usada en las VANETs para controlar y observar el	1. Development and evaluation of smartphone-based ITS applications for vehicular networks. (Escriba,

	comportamiento del vehículo a velocidad altas y en circunstancias en las que el vehículo tiende hacer un giro o cambio de ruta.	Manzoni, Veelaert, & Patra, 2019) 2. Calculating Radius of Turn from Yaw Rate. (VBOX) 3. Feature Selection for Anomaly Detection in Vehicular Ad Hoc Networks. (Le, Hartog, & Zannone, 2018)

Tabla 8. Variables adicionales
Fuente: Propia

ATTACK SYBIL


```
{
"type":3,
"rcvTime":10800.392893331087,
"sendTime":10800.392784793165,
"sender":13,
"messageID":59,
"Angle":263.4167203390052,
"Speed":30.72975873583296,
"Acceleration":0.0,
"Density":3,
"distance":96.40305004608406,
"C02":5.273115473747277,
"radius of turn":-19571806.597754215,
"Yaw Rate":9.786295588573752e-7,
"direction":[-0.1146472546276266,0.9934062648314375,0.0],
"Position":[3597.1520859538707,5542.199221013564,1.895],
"Speed2":[-3.178365760312756,38.7989694164331,0.0],
"RSSI":1.115287211193829e-7
}
```


Figura 12 Estructura del formato json del ataque sybil
Fuente: Propia

```
  ATTACK TIMING
{
"type":3,
"rcvTime":10800.473721908,
"sendTime":10800.473608040018,
"module accident":0,
"Time accident":0.0,
"sender":19,
"messageID":204,
"Angle":263.4167203390124,
"Speed":30.8254893423577,
"Acceleration":0.0,
"Density":2,
"distance":16.432900786632776,
"C02":5.302864837530882,
"radius of turn":-963969766.6310403,
"Yaw Rate":2.8108467482170194e-8,
"direction":[-0.11464725462750329,0.9934062648314518,0.0],
"Position":[3615.1220810783718,5433.756203229574,1.895],
"Speed2":[4.952912920747734,-39.99841186821371,0.0],
"RSSI":0.0000016731990867540017
}
```


Figura 13. Estructura del formato json del ataque Timing
Fuente: Propia

La cantidad de mensajes transmitidos en las simulaciones varía dependiendo de los ataques y las densidades de tráfico: en densidades bajas, se envían 7218 a 12016 mensajes, en densidades medias, hay entre 30024 y 46484 mensajes, y en densidades altas, hay 52529 a 76044 mensajes enviados. En la Figura 14 se muestra los mensajes (*beacons*) versus las densidades de tráfico.

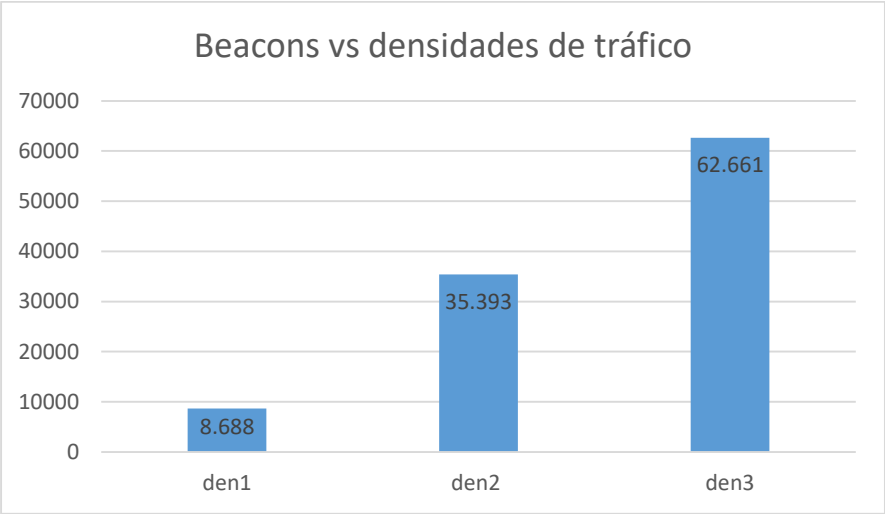

Figura 14. Beacons vs densidades de tráfico
Fuente: Propia

3.6. Planificación

Como se ha descrito en el transcurso del documento este proyecto simula dos tipos de ataque, cada uno de ellos ejecuta una cantidad de simulaciones en diferentes densidades de tráfico, con probabilidades distintas y con un número de repeticiones determinada. En la Tabla 9 se detalla la planeación de los experimentos de simulación.

	Probabilidades de atacantes						
Densidad	10%	20%	30%	Repetición	Preparación de la simulación	Tiempo de simulación	Duración de simulación
Baja (3h)	0.1	0.2	0.3	5	21 min	600 s	4h 35 min
Media (5h)	0.1	0.2	0.3	5	40 min	600 s	8h 56 min
Alta (7h)	0.1	0.2	0.3	5	1h 20 min	220 s	15h 25 min

Tabla 9. Detalles planificación
Fuente: Propio

En resumen, en este capítulo se observa cómo crear e implementar un dataset con los simuladores SUMO, OMNeT y VEINS, estudiando ataques cibernéticos e interpretando cuales variables son las más utilizadas a la hora de la detección de estos ataques. Todo esto con el fin de que este dataset sirve como base para investigaciones en la que se comparen algoritmos de detección de mala conducta sobre estos ataques y aumentar la información del

proyecto VeReMi con nuevos ataques y variables. Sin embargo, estos datos también sirven para otras temáticas que permitan tomar enfoques relacionados con big data, con la seguridad en la construcción de los vehículos, con la protección del medio ambiente y entre otras, que permitan identificar y solucionar problemas que van apareciendo en el transcurso de la evolución de la VANETs.

4. VALIDACIÓN DEL DATASET

En este capítulo se valida el dataset obtenido que consta de dos paquetes, cada uno representa un ataque diferente (ataque Sybil y Timing) donde en ellos se guardan los 45 resultados de las simulaciones realizadas por el ataque correspondiente. Cada resultado obtenido por simulación guarda dos tipos de estructura: el primer tipo es un conjunto de archivos individuales que contienen la información de mensajes recibidos por cada vehículo, donde se muestra el comportamiento de los mensajes de los vehículos atacantes como de los legítimos y a su vez las variables especificadas en la sección (3.5). En segundo lugar, un archivo de verdad donde se guarda los mensajes recibidos de todos los nodos, en este archivo se observan todas las variables especificadas en la sección (3.5) y también se puede verificar la existencia de ciertos mensajes modificados en los archivos de primer tipo. Este dataset obtenido por el conjunto de simulaciones de cada ataque se valida, detectando el mal comportamiento de los mensajes de cada vehículo mediante la detección de plausibilidad utilizando cuatro mecanismos distintos. Como se ha dicho previamente este trabajo se basa en el proyecto VeReMi donde se usan estos mismos mecanismos de plausibilidad para la detección de mal comportamiento, a su vez se implementó una réplica de su validación de manera básica. Ya que el objetivo principal de este proyecto es la creación de un dataset con dos ataques diferentes simulados. En la Tabla 10 se muestras las herramientas utilizadas en la validación.

Herramientas de validación	
Python	Lenguaje que facilita la codificación
Anaconda	Es una distribución de Python que funciona como un gestor de entorno, un gestor de paquetes y que posee una colección de más de 720 paquetes de código abierto**.**
Librerías	Pandas, Json, Matplotlib, CSV, Math.
Pycharm	Entorno de desarrollo de Python y otros lenguajes

Tabla 10. Herramientas de validación
Fuente: Propia

4.1. Validaciones en Python

En esta sección se describe el paso a paso en forma general de la codificación de las validaciones en lenguaje Python:

1. Se interpreta la codificación del proyecto VeReMi y se instalan las librerías necesarias para la implementación en lenguaje Python, con la distribución Anaconda.
2. Se analiza la estructura de los archivos json del dataset. Para determinar cómo codificar la lectura de estos y así recorrerlos, para utilizar las variables necesarias en la implementación.
3. Se estudia el comportamiento de los mecanismos de plausibilidad del proyecto VeReMi que son: el umbral de rango de aceptación (ART), la advertencia de aparición repentina (SAW), la verificación de velocidad simple (SSC) y la verificación de la distancia recorrida (DMV), con el fin de entender cada uno y poder codificarlos de la manera más adecuada. Permitiendo un mejor manejo y comprensión de las variables utilizadas.
4. Se analiza el comportamiento del rendimiento de análisis de datos utilizado en la validación del proyecto VeReMi, para implementarlo en el algoritmo.
5. Los resultados del rendimiento del análisis de datos se guardan en una lista, con el fin de almacenarlos en un archivo con formato CSV. Para graficar el comportamiento de los mecanismos.
6. Al realizar todos los pasos anteriores se replica la validación de VeReMi, la cual se denomina validación 1. Ahora se modifica esta validación agregándole a los dos mecanismos de plausibilidad (ART y SAW), dos reglas nuevas de detección una para el ataque *Sybil* y otra para el ataque *Timing* generando un nuevo archivo denominado validación 2.
7. Por último, al observar el comportamiento de los mecanismos por nodo, se modificaron las dos validaciones creadas para que puedan evaluar el comportamiento por simulaciones.

4.2. Evaluando la calidad de la detección

En esta sección se establece la métrica con la que se decide la calidad del detector, la cual se basa en la matriz de confusión (que básicamente corresponde a una descripción de los falsos positivos/negativos y verdaderos positivos/negativos) y para manejar el rendimiento de análisis de datos, es mejor proporcionar dos valores cuantificables para mostrar la eficacia de este. Entonces se toma como formulación, el cálculo de precisión de recuperación: la precisión es la fracción de todas las instancias relevantes dividido entre las instancias obtenidas. Y la recuperación es la fracción de instancias relevantes que se han obtenido sobre la cantidad total de instancias relevantes.

4.3. Evaluación de detectores de plausibilidad

El proyecto VeReMi permite utilizar su código abierto ya sea para las simulaciones y en este caso para la validación, donde muestra una aplicación para el dataset obtenido y métricas para detectores de plausibilidad centrado en los datos. Estos detectores son establecidos en un marco de lógica subjetiva. En este trabajo se compararon las cuatro mecanismos de detección: ART, SAW, SSC y DMV. De estos, el ART es el más estudiado, usando el rango de recepción esperado como una medida de la plausibilidad de la posición incluida en los mensajes de BSMs entrantes de un solo salto, que son la fuente de información más importante para las aplicaciones de seguridad en VANETs. La advertencia de aparición repentina se basa en el supuesto de que los vehículos no aparecerán repentinamente, sino que siempre se aproximarán desde la distancia, sí un mensaje se origina cerca de un remitente desconocido, se considera malicioso. La verificación de velocidad simple decide el nivel de mal intención según la relación entre la velocidad reclamada y la velocidad implícita en las diferencias de posición y tiempo entre el mensaje actual y el anterior, y la velocidad reclamada en el mensaje actual. Sí la desviación supera un umbral, este detector clasifica el mensaje como malicioso. Finalmente, la verificación de distancia recorrida, verifica sí el vehículo se movió una distancia mínima y sí esta distancia es demasiado pequeña, el mensaje se considera malicioso (Heijden, Lukaseder, & Kargl, 2018). En la Tabla 11 se muestra los mecanismos de detección de plausibilidad y los umbrales en los que se validan. Cada

conjunto de valores (umbrales) de los mecanismos de detección de la tabla son parámetros mínimos que proporcionan una estimación de la precisión de recuperación con la que se detectan los ataques cibernéticos. También me permitió identificar cuál de estos umbrales son los más adecuados para la detección de cada mecanismo de plausibilidad.

Detector	Parámetros	Valores
ART	reception range(m)	100,200, 300, 400,450,500,50,600,700,800
SAW	max. Appearance distance(m)	25,100,200
SSC	max. Speed deviation (m/s)	7.5,10,15,20,25,30
DMV	min. Distance(m)	10,15,20,25

Tabla 11. Parámetros de detección
Fuente: Proyecto VeReMi

4.4. Rendimiento de detección

En la siguiente sección se muestra la comparación de las dos validaciones implementadas y especificadas en la sección 4.1 en donde individualmente registran los resultados del análisis de detección de plausibilidad. Para una explicación más sencilla de los resultados, se tomó el escenario de baja densidad y probabilidad de 30%. En primer lugar, se utilizaron los parámetros de detección de la Tabla 11, en la validación de diferentes nodos donde su simulación cumplía con el escenario previamente dicho, de tal manera que se puede reflejar la interacción de estos mecanismos en diferentes umbrales y especificar concretamente el resultado de cada uno. Por otra parte, al ver la interacción de la validación por nodos se buscó unificar la validación por simulaciones, para esto se deja en los mecanismos un umbral fijo que es el más eficiente en la detección de cada uno.

Ahora se hará una breve discusión de las dos validaciones realizadas para cada ataque, donde se busca describir el comportamiento de los mecanismos de plausibilidad y así determinar cuál de estos es más preciso para detección de cada ataque.

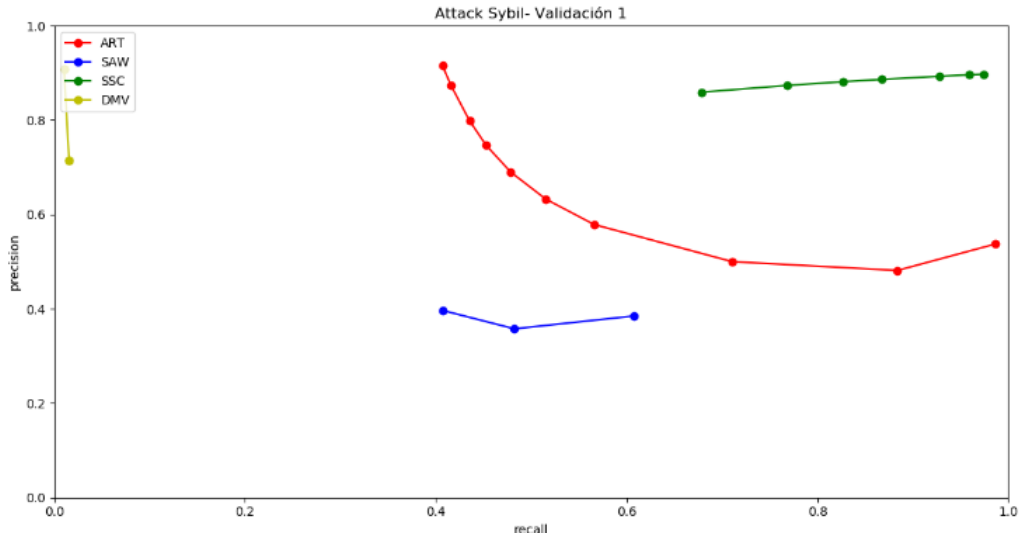

Figura 15. Attack Sybil- validación 1
Fuente: Propia

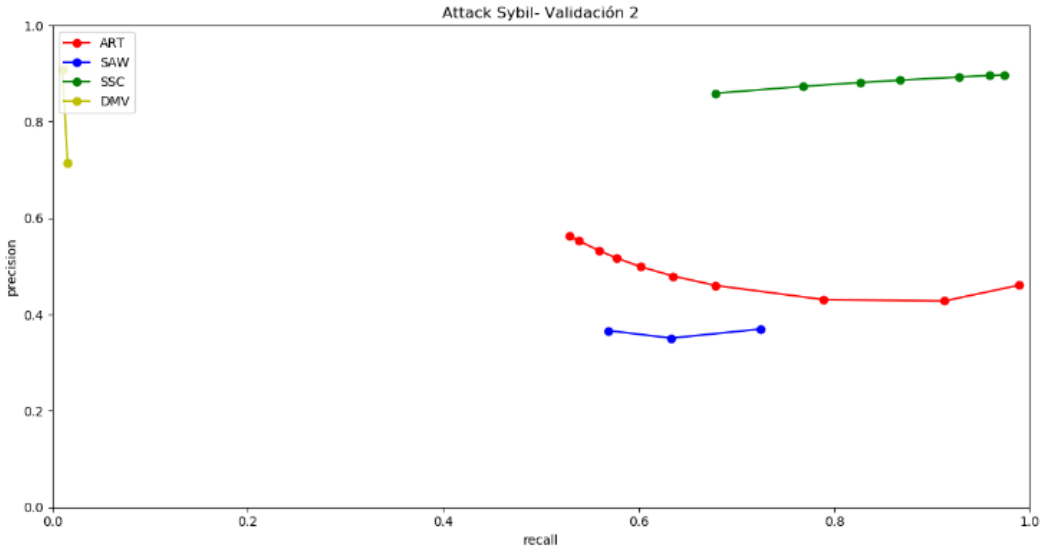

Figura 16. Attack Sybil- validación 2
Fuente: Propia

Para el ataque *Sybil* en las dos validaciones, Figuras 15 y 16, los resultados muestran que en densidad baja este ataque es difícil de detectar para todos los mecanismos. Ya que cambian su precisión de recuperación dependiendo del algoritmo de validación a excepción de los detectores DMV y SSC que se mantienen constantes. Con respecto al SSC, que verifica sí la velocidad reclamada en el mensaje actual corresponde a la velocidad implícita entre la posición del mensaje actual y el mensaje anterior, esta detección no es la apropiada para este ataque porque el comportamiento de los vehículos permanece natural. A su vez el DMV no funciona en absoluto ya que el atacante agrega el mismo valor a cada mensaje causando la

misma situación del detector SSC. El ART por otro lado, se observa que en umbrales mayores de 500 su detección es peor, pero en las validaciones se determina que para la validación 2 aumenta su detección ya que se mejoró el mecanismo para detectar mensajes fantasmas y así sucede con el detector SAW, aunque a diferencia de los otros, este es muy importante ya que su implementación se realizó para este tipo de ataques.

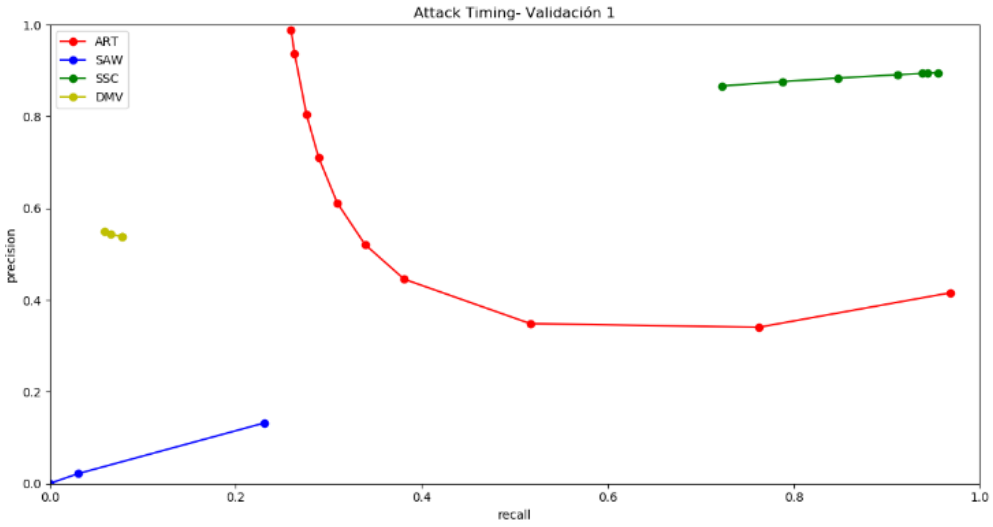

Figura 17. Attack Timing- validación 1
Fuente: Propia

Para el ataque *Timing* en las dos validaciones, Figuras 17 y 18 los resultados muestran que en densidad baja este ataque es difícil de detectar al igual que el anterior para todos los mecanismos, ya que cambian su precisión de recuperación dependiendo del algoritmo de validación a excepción de los detectores DMV y SSC que se mantiene constantes. Con respecto a estos dos últimos su comportamiento es similar al del ataque *Sybil* a diferencia que el DMV es mucho más impreciso en este ataque. A su vez el ART tiene el mismo comportamiento que el anterior. Por otro lado, el detector SAW aumento considerablemente en este ataque por el retraso de los mensajes y el aumento de la recuperación de mensajes en la recepción y transmisión de ellos.

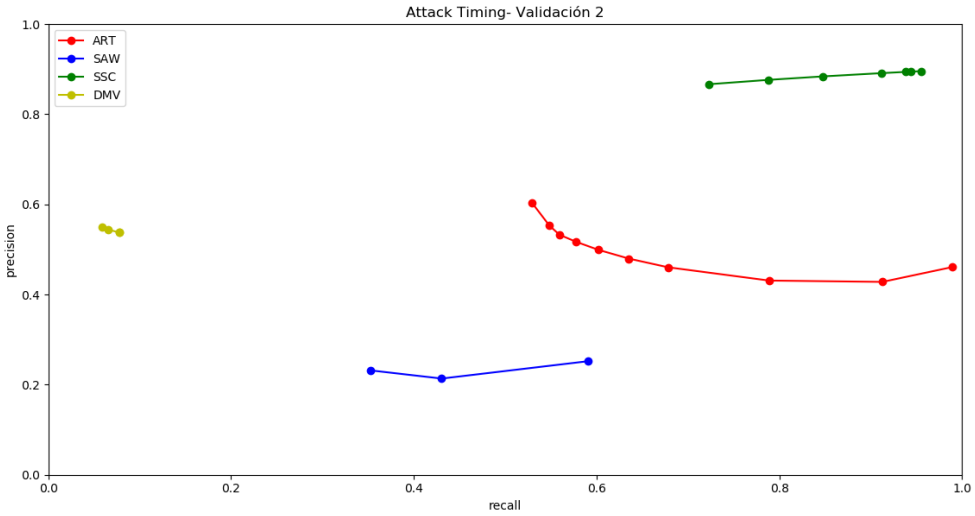

Figura 18. Attack Timing- validación 2
Fuente: Propia

Como se explicó al comienzo de esta sección se mejoraron las validaciones para que fueran por simulaciones, para ello se determinó para cada mecanismo de detección un umbral específico. Para el ART un umbral de 400 que es más efectivo en la precisión de los ataques, para SAW un umbral de 200 que permite detectar los mensajes fantasmas de mejor manera, para SSC un umbral de 30 que es lo normal en carretera y para DMV un umbral de 20. Posteriormente se muestra los ataques y validaciones con esta implementación.

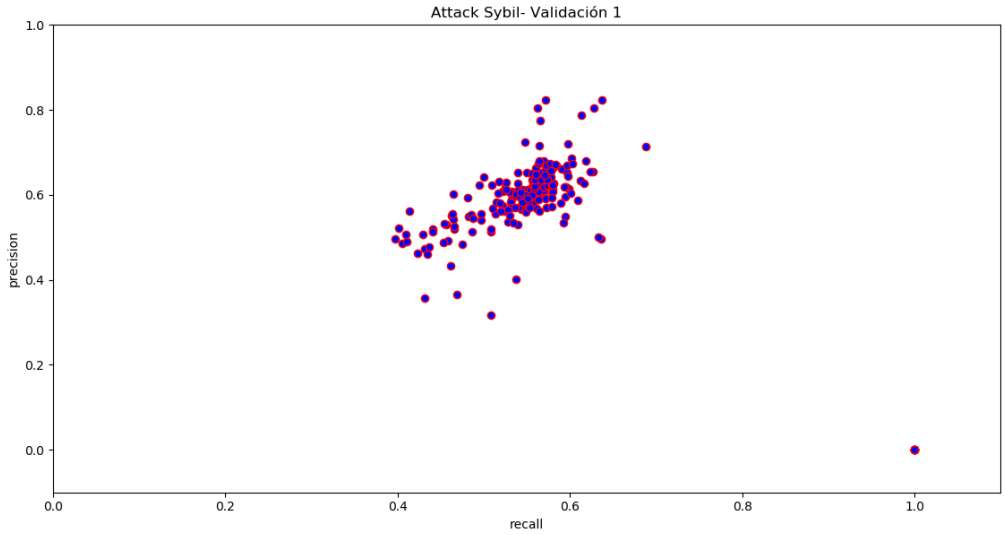

Figura 19. Attack Sybil- validación 1 implementada
Fuente: Propia

En la Figura 19 se muestra el ataque *Sybil* validación 1 en una gráfica de dispersión. Cada uno de los puntos representa un nodo analizado en los cuatro mecanismos de plausibilidad, llevando a determinar la precisión y recuperación de cada uno y evidenciar la validación por simulación. A demás se puede observar que el ataque es difícil de detectar porque está por debajo de los índices de detección idóneos (1:1), aunque en algunos nodos tienden a aproximarse a la detección idónea y así determinarlos como atacantes.

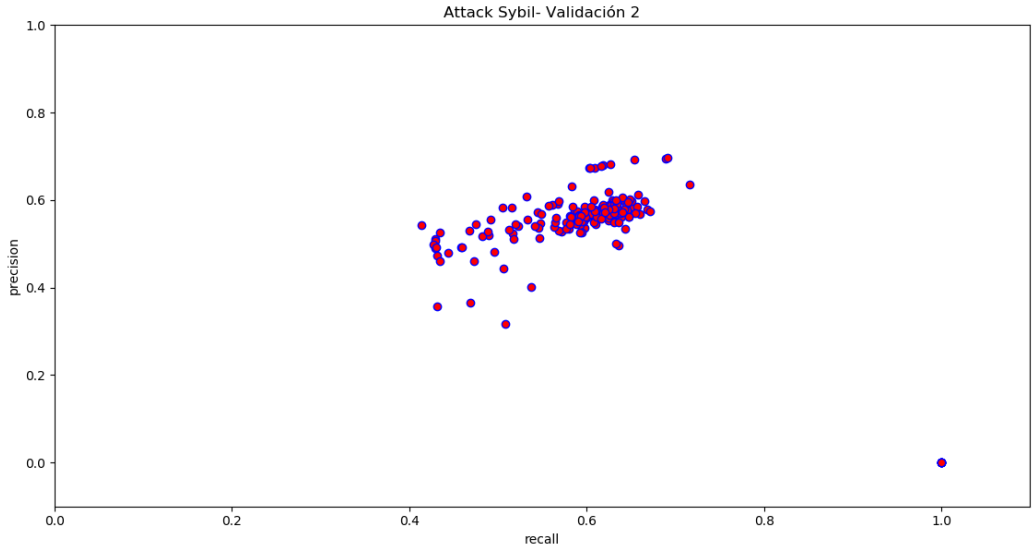

Figura 20. Attack Sybil- validación 2 implementada
Fuente: Propia

En la Figura 20 se muestra el ataque *Sybil* validación 2 en una gráfica de dispersión. Al igual que la anterior grafica se describe por nodos y analizados por los mismos mecanismos de plausibilidad. A demás se puede observar que el comportamiento de la detección del ataque es similar ya que está por debajo de los índices de detección idóneos (1:1), aunque se muestra un aumento en la recuperación de mensajes de recepción y transmisión que aproxima a la detección idónea del ataque. Determinando que la validación 2 es mejor detector que la primera.

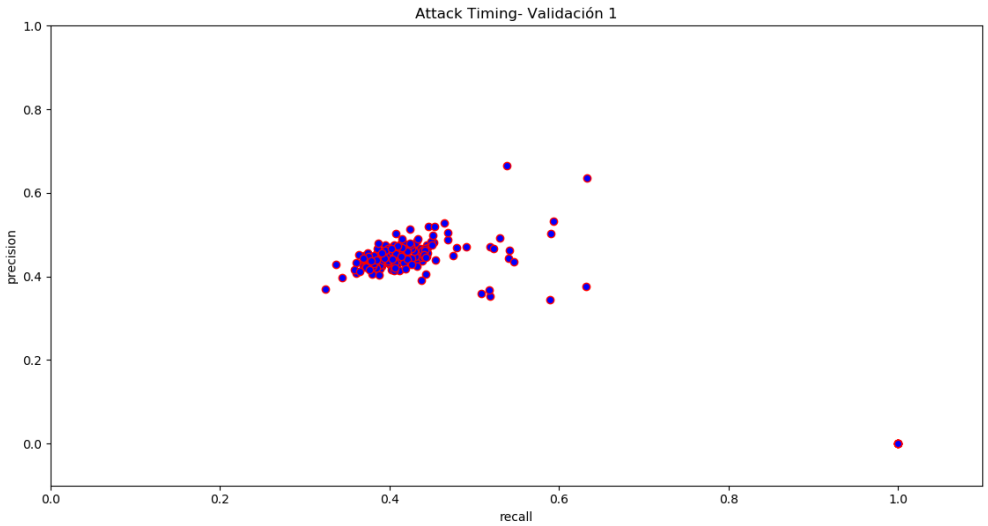

Figura 21. Attack Timing - validación 1 implementada
Fuente: Propia

En la Figura 21 se observa el ataque *Timing* validación 1 en una gráfica de dispersión. Cada uno de los puntos representa un nodo analizado en los cuatro mecanismos de plausibilidad. Por otro lado, la gráfica muestra que la detección de este ataque está por debajo de los índices idóneos y a su vez determinamos que la detección para este ataque no es válida por la falta de precisión y recuperación.

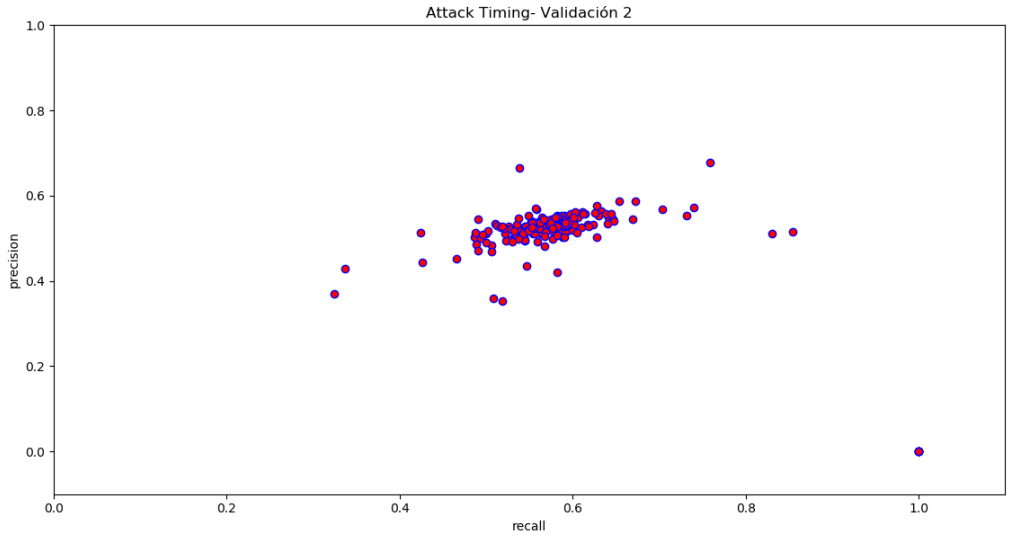

Figura 22. Attack Timing - validación 2 implementada
Fuente: Propia

En la Figura 22 se observa el ataque *Timing* validación 2 en una gráfica de dispersión. Cada uno de los puntos representa un nodo analizado en los cuatro mecanismos de plausibilidad. Por otro lado, la gráfica muestra que la detección de este ataque está por debajo de los índices idóneos y a su vez un aumento notorio de la precisión de recuperación de mensajes de recepción y transmisión, en comparación a la validación 1 de este ataque. Por esto se puede determinar que la validación 2 es aceptables para este ataque.

5. CONCLUSIONES

- En este proyecto se logró obtener un conjunto de datos como se pretendía con ayuda de los simuladores (SUMO, OMNeT y VEINS) y con la interpretación concreta del comportamiento de los ataques cibernéticos implementados (ataque Sybil y Timing). Además, se agregaron nuevas variables con respecto al dataset del proyecto VeReMi (descritas en la tabla 8), con el propósito de que amplié la información de este proyecto y dar una base sobre la cual los investigadores puedan hacer comparaciones de los resultados obtenidos de diferentes algoritmos de detección en distintas densidades de tráfico e implementaciones de atacantes.
- Las aplicaciones de seguridad constituyen un importante enfoque en la VANETs, ya que su objetivo principal se centra en evitar las congestiones de tráfico, para ayudas de emergencia y para prevenir accidentes. Mediante la simulación de las VANETS es posible definir parámetros necesarios para el desarrollo de este tipo de aplicaciones, siguiendo perspectivas y mecanismos que permitan evitar vulnerabilidades en la comunicación entre vehículos.
- La simulación de las VANETs a lo largo de los años se ha convertido en el enfoque más utilizado para estudios e investigación de diferentes temáticas en el entorno de los ITS, por su desarrollo en implementaciones de aplicaciones y sistemas cada vez más similares a las problemáticas de la vida real de tráfico vehicular.
- Se mostró la aplicación del dataset con cuatro mecanismos de detección de plausibilidad (el umbral del rango de aceptación, la advertencia de aparición repentina, verificación de velocidad simple y verificador de distancia), y se proporcionó una discusión del cálculo de la recuperación de precisión descrita en la sección 4.4, estableciéndolo como un método que permite al investigador determinar que detectores puede mejorarse potencialmente para así tener una mejor detección de los ataques y de confianza en los mensajes. Por otro lado, este tipo de mecanismos son ideales para detectar ataques sybil en especial el de advertencia de aparición repentina ya que este mecanismo facilita la detección de mensajes fantasmas.

6. REFERENCIAS BIBLIOGRÁFICAS

Ana María Orozco, G. L. (2012). Redes vehiculares Ad-hoc: aplicaciones basadas en simulación.

Campos, J. M., & Reina, D. G. (2014). *Evaluación de Protocolos de Encaminamiento Para Redes Vehiculares (VANET).* sevilla.

Escriba, D. J.-C., Manzoni, D. P., Veelaert, D. P., & Patra, S. (2019). *Development and evaluation of smartphone-based ITS applications for vehicular networks.* VALENCIA, españa.

Eziama, E., Tepe, K., Balador, A., Nwizege, K. S., & Jaimes, L. M. (2018). Malicious Node Detection in Vehicular Ad-Hoc Network Using Machine Learning and Deep Learning. *IEEE Globecom Workshops (GC Wkshps)*.

García, O. A., & Águila, M. S. (2016). Evaluación del desempeño de la arquitectura de red WAVE en entornos vehiculares de alta velocidad y densidad de nodos.

GIL, C., & GIL, P. C. (2012). *Soluciones para la autenticación y gestión de subredes en MANETs y VANETs.*

GONZÁLEZ, P., & CEREZO, R. (2012). ESTUDIO DEL SIMULADOR DE REDES VEHICULARES VEINS.

Grover, J., Kumar, D., Sargurunathan, M., Gaur, M., & Laxmi, V. (2010). Performance Evaluation and Detection of Sybil Attacks in Vehicular Ad-Hoc Networks.

Heijden, R. v., Dietzel, S., & Kargl, F. (2014). Misbehavior Detection in Vehicular Ad-hoc Networks.

Heijden, R. W., Lukaseder, T., & Kargl, a. F. (2018). VeReMi: A Dataset for Comparable Evaluation of Misbehavior Detection in VANETs.

Hussain, R., Kim, S., & Oh, H. (2012). Privacy-Aware VANET Security: Putting Data-Centric Misbehavior and Sybil Attack Detection Schemes into Practice.

J., L. M., & Moreira, E. (2019). An evaluation of reputation with regard to the opportunistic forwarding of messages in VANETs.

Joe, M. M., & Ramakrishnan, B. (2016). Review of vehicular ad hoc network communication models including WVANET (Web VANET) model and WVANET future research directions.

Junaid, M. A., A, S. A., Warip, M. N., Azir, K. N., & Romli, N. H. (2018). Classification of Security Attacks in VANET: A Review of Requirements and Perspectives.

Kukreja, D., Dhurandher, S. K., & Reddy, B. V. (2017). Power aware malicious nodes detection for securing MANETs against packet forwarding misbehavior attack.

Le, V. H., Hartog, J. d., & Zannone, N. (2018). Feature Selection for Anomaly Detection in Vehicular Ad Hoc Networks.

Nyangaresi, V. O., Abeka, D. S., & Rodgrigues, P. A. (2018). Secure Timing Advance Based Context-Aware Handover Protocol for Vehicular Ad-Hoc Heterogeneous Networks.

O. A. Orozco, D. F., & Calderón, O. J. (2014). *Impacto de la velocidad y modelo de movilidad en una comunicación de datos de una red vehicular*.

Orozco, O. A., & Gonzalo Llano, P. (2014). *OSA: A Vanet application focused on fuel saving and reduction of CO2 emissions.* cali,colombia.

Reu2015.weebly. (1 de 11 de 2019). *reu2015.weebly*. Obtenido de reu2015.weebly: https://reu2015.weebly.com/background.html

Sakiz, F., & Sen, S. (2018). A survey of attacks and detection mechanisms on intelligent transportation systems: VANETs and IoV.

Sánchez, J. A. (2017). Redes Vehiculares Aplicadas a la Movilidad Inteligente y Sostenibilidad Ambiental en Entornos de Ciudades Inteligentes.

Sarasti, O. O., & Ramírez, G. L. (2014). APLICACIONES PARA REDES VANET ENFOCADAS EN LA SOSTENIBILIDAD.

Sedano, D. P., & Gil, R. S. (2019). *Evaluación de protocolos de enrutamiento en redes ad-hoc de vehículos (VANET).*

Sommer, C. (2006). *Veins*. Obtenido de Veins: https://veins.car2x.org/

SUMRA, I. A., MANAN, J.-L. A., & HASBULLAH, H. (2014). Timing Attack in Vehicular Network.

Vasudevaa, A., & Sood, M. (2018). Survey on sybil attack defense mechanisms in wireless ad hoc networks.

VBOX, R. (s.f.). *Calculating Radius of Turn from Yaw Rate.*

Yua, B., Xu, C.-Z., & Xiao, B. (2013). Detecting Sybil attacks in VANETs.

Anexos

Textos de referencia para futuros estudios

Para el correcto entendimiento de este libro en la parte de simulaciones quisiera facilitar a los interesados un proyecto que fue de utilidad a la hora de interpretar cada uno de los simuladores. Sin embargo, no fue el único en que se basó el estudio de estos simuladores ya que en cada uno de los proyectos referenciados en la elaboración del libro contienen temática de estos.

1. ESTUDIO DEL SIMULADOR DE REDES VEHICULARES VEINS (PABLO GONZÁLEZ-RIPOLL CEREZO)

   Este proyecto contiene una incursión en profundidad en el mundo de las redes vehiculares (VANETs), las analiza conceptualmente mostrando su potencial y determinando que las herramientas de simulación son un medio importante para su desarrollo. A demás se enfoca en la terminología de los componentes de VEINS (OMNeT++, SUMO y el módulo TRACI esencialmente) y aportar un estudio sobre las funcionalidades que ofrecen cada uno de estos simuladores. En el siguiente link se encuentra el proyecto https://e-archivo.uc3m.es/bitstream/handle/10016/16704/PFC_PabloGonzalez-RipollCerezo.pdf?sequence=1.

Para instalación de los simuladores ingresar en el siguiente link donde especifica el paso a paso de las instalaciones. https://veins.car2x.org/tutorial/