Authentication and consensus overhead in vehicular ad hoc networks

Jonathan Petit, Zoubir Mammeri

Authentication and consensus overhead in vehicular ad hoc networks

Jonathan Petit · Zoubir Mammeri

Abstract Vehicular ad hoc networks aim at increasing passenger safety by exchanging warning messages between vehicles wirelessly. A main challenge is to resist to various malicious abuses and security attacks. However, any security mechanism comes with overhead. We analyze how the authentication algorithm ECDSA and the consensus mechanism impact the vehicular network performance and the braking distance. Processing and communication overheads, decision methods for consensus, are analyzed by analytical models and intensive simulations. We propose a formula to assess the total time overhead of the authentication. Results conclude that the authentication key size should be chosen carefully, and the decision method should be adapted to the context.

Keywords Authentication overhead · Vehicular networks · ECDSA · Consensus

1 Introduction

In 2007, road accidents have cost 110 deaths, 4600 injuries and €438 millions daily in the European Union [1]. The damage is similarly devastating in the United States with 102 deaths, 7900 injuries and $630 millions daily [2, 3]. Therefore, industry consortia, governments, and automotive companies, have made the reduction of vehicular fatalities a top priority [4]. To raise this challenge, a main idea is to make vehicles and roads smarter thanks to wireless communications. Indeed, wireless communications will increase the line-of-sight of the driver and make vehicles aware of their environment. Modern vehicles now include a set of processors connected to a central computing platform that provides many wired and wireless interfaces. Smart vehicles are those vehicles that are equipped with On-Board Unit (OBU), which has recording, processing, positioning, and location capabilities and that supports wireless security protocols. Roads can be made smart with Road-Side Units (RSU), installed along a road, that can inform passing vehicles about the road traffic conditions. A wireless vehicular network is composed by OBU and RSU, connected wirelessly. The VSC Project [5] details the 75 applications that could be deployed on vehicular networks. Applications are divided in three categories: safety-related, traffic optimization and infotainment. Automotive safety-related applications aim to assist drivers in avoiding vehicular accidents, by providing advisories and early warnings to drivers, using broadcast vehicle-to-vehicle (V2V) communications. Vehicles typically communicate as per the Dedicated Short Range Communication standard (DSRC) [6], and broadcast messages in response to certain notified events (emergency message) or periodically (beacon message) [7]. In this paper, we focus on V2V communications in Local Danger Warning (LDW) application, which is considered one of the most promising active safety applications for inter-vehicle communication [8]. Since drivers of vehicles participating in V2V communications are expected to act on messages received from other participants, it is clearly necessary that these messages be transmitted in a secure fashion. Unfortunately, security mechanisms come with overhead that impact the performance of the V2V communications, and hence that of the safety applications. The IEEE 1609.2 standard

J. Petit (✉) · Z. Mammeri
IRIT, Université de Toulouse, 118 route de Narbonne,
31062 Toulouse cedex 9, France
e-mail: petit.jonathan@ieee.org

Z. Mammeri
e-mail: Zoubir.Mammeri@irit.fr

for vehicular ad hoc networks is based on the ECDSA algorithm for supporting the authentication mechanism. The main goal of this work is to define a formula, which assesses the authentication overhead in VANET. We also introduce the problem of consensus, which is an additional mechanism that impacts the total time overhead of ECDSA.

The paper is organized as follows. First, we survey previous works. The authentication mechanism ECDSA is discussed in Sect. 3. Then, we investigate the processing and communication overhead of ECDSA in Sects. 4 and 5. In Sect. 6, we define the global authentication overhead of ECDSA. Section 7 introduces the problem of consensus and analyzes decision methods. Extensive simulations are done to evaluate the impact of ECDSA on the braking distance. A discussion highlights limits of our work and proposes optimizations in Sect. 8. Section 9 concludes the paper.

2 Related work

In [9], Iyer et al. provided an evaluation of the computational overhead in V2V communications. They observed that the performance bottlenecks could shift from security layer to MAC layer, depending on the system parameters. They provided interesting values like buffer size at MAC layer. But their work is independent of security protocols and computational capabilities. It does not give results about ECDSA overhead. Moreover, they did not analyze the communication overhead.

Haas et al. [10] performed simulation using real vehicle mobility from I-80 in Emeryville, California, United States. They compared ECDSA (with P-224) and TESLA, analyzing the communication range and the MAC layer delay. Moreover, they provided an assessment of verification latency for various hardware configurations. Their simulation results show that TESLA performs poorer than ECDSA but has a lower latency because of the smaller packet size.

In [11], Rao evaluated the performance of certification mechanisms in vehicular networks. He analyzed the certificate distribution and revocation using Certificate Revocation Lists (CRLs) and Freshness Checks. Moreover, the impact of a queue at the security layer was investigated. Our work complete this work by investigating the ECDSA overhead.

Our work differs from the above-mentioned studies. First, we investigate the impact of the authentication key size. Then we translate the overhead into an issue—namely braking distance—of safety application like Local Danger Warning.

Ostermaier et al. proposed the voting schemes to face the problem of consensus in vehicular networks [8]. They analyzed four decision methods to evaluate the plausibility of received hazard messages. Their simulation results show that the "*majority of Freshest X with Threshold*" is the best efficient method to avoid false warning injection attack. Section 7 completes this work by addressing the issue of setting dynamically *X* and *Threshold* parameters.

3 Authentication in vehicular networks

In order to secure vehicular communications, Wireless Access in Vehicular Environments (WAVE) architecture mandates the use of PKI mechanisms, where service application messages are encrypted and vehicle safe-ty messages are digitally signed. All implementations of IEEE 1609.2 standard [12] shall support the Elliptic Curve Digital Signature Algorithm (ECDSA) [13] over the two NIST curves P-224 and P-256. Moreover, all implementations of this standard that support encryption shall support the Elliptic Curve Integrated Encryption Scheme (ECIES) over the NIST curve P-256. ECDSA provides authentication with digital signature utilizing the elliptic curve discrete logarithm problem (ECDLP) as the underlying untraceable operation.

3.1 ECDSA algorithm

ECDSA is a variant of the Digital Signature Algorithm (DSA), which operates on elliptic curve groups. In [6, 14], Neal Koblitz and Victor S. Miller introduced the use of elliptic curves in cryptography (ECC). ECC is an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite fields. To use ECC all parties must agree on the elements defining the elliptic curve, which are domain parameters of the scheme. Each participant does not usually achieve the generation of domain parameters since this involves counting the number of points on a curve, which is time-consuming and troublesome to implement. As a result, NIST and SECG published domain parameters of elliptic curves for several common field sizes [7, 15]. Johnson signature scheme [16] is an algorithm to compute ECDSA, and includes three phases: key generation, signature generation and signature verification. These phases are described in Algorithm 1.

3.2 Complexity of ECDSA

In this section, we investigate the time complexity of scalar multiplication, modular multiplication and inversion, to evaluate the ECDSA processing time and complexity.

3.2.1 Scalar multiplication

In ECDSA, a scalar multiplication [17] of a given random point is used in signature generation and verification. This

Algorithm 1 ECDSA

```
a. Key generation
1. Obtain a set of elliptic curve domain parameters
$q$ denotes the size of the underlying field $F_q$, which can be a large prime or a prime to a power
$a, b \in F_q$: parameters of elliptic curve $E$
$G \in E$: point on $E$
$n$: order of $G$ ($n$ prime greater than 2160)
$h = ord(E)/ord(G)$, where $ord(X)$ denotes the order of $X$
2. Select a random number $d \in [1, n-1]$ as private key
3. Compute the public key $Q = dG$
$(E, Q, G, n)$ are public.

b. Signature generation
input: message $m$ and $(d, Q)$
1. Select a random number $k \in [1, n-1]$
2. Compute $kG = (x_1, y_1)$ and $r = x^{-1}$ mod $n$. If $r = 0$ goto 1
3. Compute $s = k^{-1}(e + dr)$ mod $n$ with $e = H(m)$. If $s = 0$ goto 1
4. The signature of $m$ is $(r, s)$.

c. Signature verification
input: $(r, s), m, Q$
1. Verify $r, s \in [1, n-1]$
2. Compute $w = s^{-1}$ mod $n$
3. Compute $u_1 = ew$ mod $n$ and $u_2 = rw$ mod $n$ with $e = H(m)$
4. Compute $X_1 = u_1G + u_2Q$ and $V = X_1$ mod $n$
5. If $V = r$ then signature accepted.
```

operation is the most time-consuming part of the total signature computation [18]. Specifically, given a n-bit long scalar k and a point P on the curve, we have to compute the elliptic curve scalar multiplication kP. There are two possible algorithms to calculate kP; the Add-and-Double algorithm and the Montgomery algorithm. These algorithms require several field multiplications, additions, and potentially one inversion of the point coordinates. Authors in [19] showed the advantage of the cryptographic usage of Montgomery-form elliptic curves in constrained environments such as mobile devices. Hence, the efficiency of ECDSA relies deeply on the arithmetic of the curve and the arithmetic of the underlying field [18].

3.2.2 *Modular multiplication*

Modular multiplication is typically the most critical operation in the computation of elliptic curves scalar multiplication. Given a word length of n bits, an n-bit integer m called the modulus, and two n-bit operands x and y, the problem is the computation of xy mod m. Montgomery's algorithm for modular multiplication [20] is considered as the fastest algorithm when x, y, and m are large [19]. The idea of Montgomery is to reduce the length of the intermediate results to a fixed quantity of $n+1$ bits. This is achieved by interleaving the computations and additions of new partial products with divisions by 2, each of them reducing the bit-length of the intermediate result by one. Algorithm 2 is used to determine a modular multiplication (n is the bit-length of x, y, or m, and b the radix) [21]. If operations (modular addition, modular multiplication, multiplication, addition, shift) on A are considered done in constant time, the time complexity is $O(n)$ and the space complexity is $O(n)$ [22, 23].

Algorithm 2 Montgomery multiplication

```
**Require:** $m = (m_{n-1} \dots m_1 m_0)_b$, $x = (x_{n-1} \dots x_1 x_0)_b$,
  $y = (y_{n-1} \dots y_1 y_0)_b$, with $0 \le x, y < m$,
  $R = b^n$ with $gcd(m, b) = 1$ and $m' = -m^{-1}$ mod $b$

  $A \leftarrow 0$ (Notation: $A = (a_{n-1} \dots a_1 a_0)_b$)
  **for** $i$ to $(n-1)$ **do**
    $u_i \leftarrow (a_0 + x_i y_0)m'$ mod $b$
    $A \leftarrow (A + x_i y + u_i m)/b$
  **end for**
  **if** $A \ge m$ **then**
    $A \leftarrow A - m$
  **end if**
  **return** $A$ // $A = xyR^{-1}$ mod $m$
```

3.2.3 *Modular inversion*

The modular inversion is another time consuming operation in scalar multiplication. The Montgomery inversion is a way to compute x^{-1} mod m. The Montgomery inversion is based on Montgomery multiplication algorithm. Montgomery inverse of an integer $x \in [1, m-1]$ is $j = x^{-1}b^n$ such that where m is prime and $n = \log_2 m$ is the bit-length. The time complexity of the Montgomery modular inversion is $O(n)$ [24].

3.2.4 *Time complexity*

We define addition, doubling, scalar multiplication, modular multiplication, modular inversion and squaring operations in elliptic curve groups as functions $P + Q$, $2P$, kP, *MUL*, *INV*, *SQR*. Hash function is denoted by *HASH*. T_X is the time needed to compute the operation X. In [25], authors proposed algorithms to compute $P + Q$ and $2P$. Table 1 summarizes time of elliptic curve point operations T_{P+Q}, T_{2P}, T_{kP}, with n the bit-length of multiplication operands.

ECDSA signature generation and verification are fully performed by modular multiplications, squaring, modular inverse and hash functions. So the time complexity of ECDSA is given in function of T_{MUL}, T_{SQR}, T_{INV} and T_{HASH}.

Signature generation time is:

$$\begin{aligned} T_{sign} &= 2T_{MUL} + T_{INV} + T_{kP} + T_{HASH} \\ &= (6n+2)T_{MUL} + T_{INV} + 5nT_{SQR} + T_{HASH} \end{aligned} \qquad (1)$$

Table 1 Operation times

Operation	Operation time
$P+Q$	$T_{P+Q} = T_{SQR} + 4 \times T_{MUL}$
$2P$	$T_{2P} = 4 \times T_{SQR} + 2 \times T_{MUL}$
kP	$T_{kP} = n \times (T_{P+Q} + T_{2P})$

Signature verification time is:

$$T_{verify} = 2T_{MUL} + T_{INV} + 2T_{kP} + T_{HASH}$$
$$= (12n+2)T_{MUL} + T_{INV} + 10nT_{SQR} + T_{HASH} \qquad (2)$$

T_{HASH} is member of (1) and (2). Hence, we analyze *HASH* time complexity. DSRC standard supports the use of SHA-224 and SHA-256 as hash functions. The SHA-256 algorithm is defined in the FIPS180-2 standard [26]. The time complexity of SHA-256 is $O(M \times n)$ where M is the size of the message to hash. Compared to the scalar multiplication, the modulus addition and subtraction operation cost is negligible and thus omitted. So, ECDSA could be divided into modular multiplication, modular inversion and hash function. As Montgomery's multiplication complexity is $O(n)$, Montgomery's modular inverse complexity is $O(n)$, and hash function complexity is $O(M \times n)$. Then, the total time complexity of ECDSA is $O(n) + O(M \times n)$.

4 Processing overhead of ECDSA

In [27], the generic time overhead of a message M is given as follows:

$$T_{ov}(M) = T_{sign}(M) + T_{tx}(Sign_{PrK_V}[M]) + T_{verify}(M) \qquad (3)$$

with:

- $T_{sign}(M)$: time to sign M.
- $T_{verify}(M)$: time to verify M.
- $Sign_{PrK_V}[M]$: signature of M by the sender V and includes the Certificate Authority's certificate of the signing key.
- $T_{tx}(Sign_{PrK_V}[M])$: time to transmit the signature.

In this section, we investigate the processing overhead of ECDSA. So, we are just interested in $T_{sign}(M)$ and $T_{verify}(M)$. These times correspond to the process done by each vehicle for transmission and reception of a signed message. The processing overhead is also known as the computational overhead. This overhead arises because the security-related operations such as signing and verifying, require a finite amount of time for completion at the processor of each vehicle. The computational overhead includes:

(i) Certificate Selection: Before signing a message a certificate needs to be selected from a pool of valid certificates for ensuring anonymity [27].
(ii) Signing Message: On receiving a message to be secured, the hash of the message is computed and signed (encrypted) using the private key associated with the certificate selected. Then, this signature and the certificate are sent (with the message) to the lower layers for transmission. In the case of ECDSA, the signature has two components, r and s.
(iii) Verifying Certificate: On receiving a message, the certificate is considered valid if it is not included in the CRLs present in the OBU in case of the CRL based schemes, or the certificate is considered fresh in case of the Freshness Check Scheme [11].
(iv) Verifying Message: In the case of ECDSA, the hash of the message is computed and used with the signature component s to obtain r. This value of r is compared with the one present in the signature.

The certification mechanisms are out of the scope of this paper and are investigated in [11]. In this paper, we focus on (ii) and (iv) overheads.

4.1 Experimentation model

There are three main cryptographic libraries: MIRACL, OpenSSL and Crypto++. Authors in [28] proposed a comparison and concluded that MIRACL has the best performance for operations on elliptic curves over binary fields. So, ECDSA was implemented using MIRACL and following the Algorithm 1. The benchmark was launched 500 times to smooth the interferences of the system due to the interrupts. All experimentations were done on an Intel Pentium D 3.4 GHz workstation with 1 Go RAM on a Mandriva 2008 Operating System. Table 2 shows operation times. Table 3 shows the execution times of signature generation and verification for ECDSA (using NIST curves P-224 and P-256).

4.2 Experimentation results

4.2.1 Processing delay

Vehicles have to generate a signature for each message sent and verify signature for each message received. The time required for these operations is called processing delay. ECDSA with a P-224 curve (respectively P-256) fits with an authentication key size of 224 bits (respectively 256). In Table 2, T_{kP} is almost equal to signature generation time. It confirms that scalar multiplication is the most expensive operation of ECDSA. Reporting values of Table 2 in formulas (1) and (2), confirms Table 3. Table 3, which gives T_{sign} and T_{verify}, shows that using P-256 instead of P-224 in

Table 2 Operation times on a Pentium D 3.4 GHz workstation

Key size (bit)	T_{MUL} (µs)	T_{INV} (µs)	T_{kP} (µs)	T_{HASH} (µs)
224	1.23	18.91	2468.71	8.47
256	1.39	22.01	3297.23	10.09

Table 3 Signature generation and verification times on a Pentium D 3.4 GHz workstation

Key size (bit)	Signature generation (ms)	Signature verification (ms)
224	2.50	4.97
256	3.33	6.63

the signature generation adds a time overhead of 33.2%. Using P-256 instead of P-224 in the signature verification adds a time overhead of 33.4%. Theoretical analysis of ECDSA shows a linear-time complexity depending on the key size. In Table 3, the processing delay increases when key size increases. These experimentation results validate the analytical model.

4.2.2 Distance

As we focus on the vehicular networks, we should not forget that vehicles are moving. During the generation and/or verification phase, a vehicle V covers a distance D_V (in meters) depending on its velocity v_V (in km/h). T (in milliseconds) is T_{sign}, T_{verify} or $T_{sign} + T_{verify}$.

$$D_V = \frac{1}{3600} \times v_V \times T \tag{4}$$

The expected number of vehicles equipped with the DSRC system, which are in transmission range R (in km), is defined in [29] by N_{TX}.

$$N_{TX} = 2N_L \gamma \rho R \tag{5}$$

The product $\gamma\rho$ is the density of equipped vehicles (in veh/km/lane), N_L the number of lanes. We consider the following scenario. In a highway with 6 lanes (3 in each direction) of 3 m each, we assume a uniform presence of vehicles, with an inter-vehicle space restricted by the safe driving distance $D_S = 0.56 \times v_V$ (in meters). Vehicles are mobile ($v_V = 130$ km/h) and transmit WSMs every 300 ms over a 300 m communication range. We consider a vehicle V located on a highway; V can hear $N_{TX} = 2 \times 6 \times 1 \times \frac{1000}{73} \times 0.3 \approx 49$ vehicles. V will receive 49 messages per 300 ms with a market penetration $\gamma = 1$ (100% of vehicle equipped with a DSRC-capable device). Before V can send a new message, it should be able to process

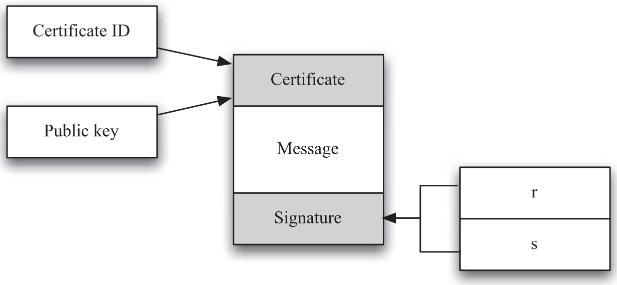


Fig. 1 Packet size overhead

all incoming messages within 300 ms. One signature verification should be faster than the maximum tolerable processing delay per message D_{PMAX}. Assuming V receives all the 49 messages, D_{PMAX} is $300/49 = 6.12$ ms. D_{PMAX} is greater than the average signature verification time for P-224. But the condition is not verified for a key size of 256 bits. So, the authentication key size should be chosen within the application constraints. According to (4), the vehicle will cover $D_V = \frac{1}{3600} \times 130 \times 6.63 = 0.239$ m during one ECDSA signature verification. Assuming that 49 messages are received at the same time, V will travel $49 \times D_V \approx 11.711$ m. The braking distance for V is defined as $D_B = \frac{(\frac{5}{18} v_V)^2}{2a}$ (in meters). In normal environment, on a dry road with a deceleration rate $a = 6.8$ m/s^2 and a velocity of 130 km/h, V stops in $D_B = 96$ m. But since V has to verify the messages, it will stop in $96 + 11.711 = 107.711$ m, corresponding at an increase of 12.2% of the braking distance. Moreover, if the vehicle does not make decisions (brake, lane change) without the driver agreement, we have to add the driver's reaction time D_R of 1.5 seconds. In spite of technologic advances in pneumatic and automotive increase driver's safety by offsetting the non-respect of D_S, the deployment of ECDSA may jeopardize these ameliorations.

5 Communication overhead of ECDSA

As we see in (3), the total time overhead of ECDSA depends on $T_{sign}(M)$, $T_{verify}(M)$, and $T_{tx}(Sign_{PrK_V}[M])$. The overhead is divided into two parts: the processing overhead and the communication overhead. After analyzing the computational overhead in the previous section, we now investigate the communication overhead $T_{tx}(Sign_{PrK_V}[M])$. The communication overhead is also known as bandwidth overhead. The bandwidth overhead arises due to the headers and footers associated with the security mechanism resulting in extra over-the-air bytes. The bandwidth overhead includes:

(i) Increased Message Size: As shown in Fig. 1, the packet transmitted over the air contains the message to be exchanged along with the certificate used to sign the message and the signature.

(ii) Dissemination of Revocation Information: In the CRL based schemes, the CRLs need to be disseminated from the infrastructure to each vehicle in the network that can potentially receive messages signed using the certificates present in the CRLs. In the case of Freshness Check scheme, the nodes have to periodically perform Freshness Check operations to ensure that their certificates are fresh.

The certification mechanisms are out of the scope of this paper. We do not analyze (ii) and focus on the (i) overhead. We also investigate the impact of the communication overhead on the transfer delay and the braking distance.

5.1 Packet overhead

The Wave Short Message format (WSM) is used for safety messages like local danger warning or periodic information message. Figure 2 describes the WSM format. Figure 3 describes the certificate format.

A WSM is first signed with ECDSA, and sent using the WAVE Short Message Protocol (WSMP). When a WSM is signed, authentication protocol adds a signed certificate (*signer*) and a signature to the original message (*unsigned_wsm*). As we see in Fig. 2, the unsigned WSM payload is 53 bytes long. The WSM header is 19 bytes long. First, when Alice signs a message with her private key, nothing guarantees that Alice can be trusted. Certificate proves this trustworthiness. A certificate of size S_{cert} (plus 1 byte for the certificate type) and a signature of size S_{sign} are appended. Figure 3 gives the OBU signing certificate format. We observe that the length of a certificate depends on two parameters:

- S_{pu} (in bits): the point size of the elliptic curve G depending on the public key algorithm associated with the key.
- $S_{sigcert}$ (in bits): the size of the signature used to sign the certificate.

The length of a certificate S_{cert} (in bytes) is defined by:

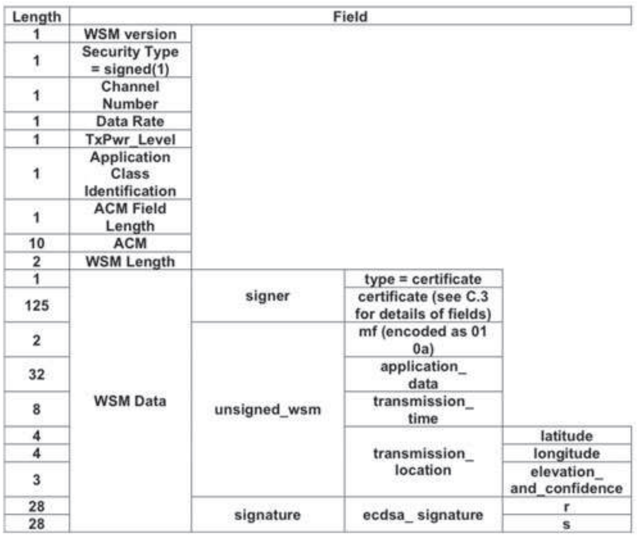
$$S_{cert} = \frac{S_{pu}}{8} + 1 + \frac{S_{sigcert}}{8} \times 2 = \frac{S_{pu}}{8} + 1 + \frac{S_{sigcert}}{4} \tag{6}$$

Secondly, to provide the authentication service, the message is signed and the signature is attached to the message. The length of a signature S_{sign} (in bytes), attached to a message, depends on the elliptic curve $S_{sigmess}$ (in bits) used in ECDSA.

$$S_{sign} = \frac{S_{sigmess}}{8} \times 2 = \frac{S_{sigmess}}{4} \tag{7}$$

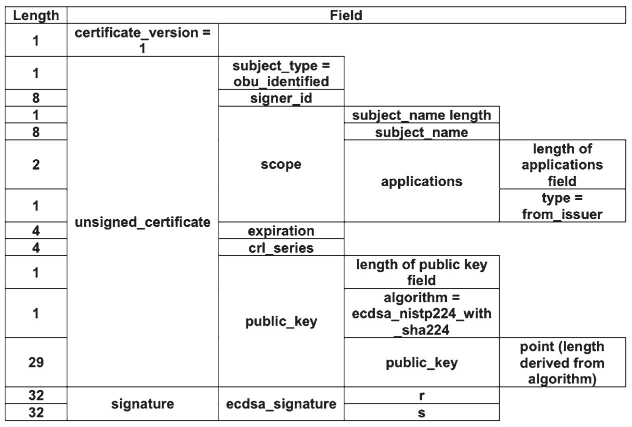
Finally, when ECDSA is used, it adds a certificate and a signature in each sent message, which results in an overhead S_{ov} (in bytes) given by the sum of (6) and (7) in formula (8):

$$S_{ov} = S_{cert} + S_{sign} = \frac{S_{pu}}{8} + 1 + \frac{S_{sigcert}}{4} + \frac{S_{sigmess}}{4} \tag{8}$$



Length	Field			
1	WSM version			
1	Security Type = signed(1)			
1	Channel Number			
1	Data Rate			
1	TxPwr_Level			
1	Application Class Identification			
1	ACM Field Length			
10	ACM			
2	WSM Length			
1	WSM Data	signer	type = certificate	
125			certificate (see C.3 for details of fields)	
2		unsigned_wsm	mf (encoded as 01 0a)	
32			application_data	
8			transmission_time	
4			transmission_location	latitude
4				longitude
3				elevation_and_confidence
28		signature	ecdsa_ signature	r
28				s

Fig. 2 WAVE safety message format [30]



Length	Field			
1	certificate_version = 1			
1	unsigned_certificate	subject_type = obu_identified		
8		signer_id		
1		scope	subject_name length	
8			subject_name	
2			applications	length of applications field
1				type = from_issuer
4		expiration		
4		crl_series		
1		public_key	length of public key field	
1			algorithm = ecdsa_nistp224_with_sha224	
29			public_key	point (length derived from algorithm)
32	signature	ecdsa_signature	r	
32			s	

Fig. 3 OBU signing certificate format [30]

The total length of a WSM (in bytes) is defined in (9):

$$\begin{aligned} S_{WSM} &= \frac{S_{sigmess}}{8} \times 2 + \frac{S_{sigcert}}{8} \times 2 + \frac{S_{pu}}{8} + 1 \\ &\quad + 32 + 20 + 53 \\ &= \frac{S_{sigmess}}{4} + \frac{S_{sigcert}}{4} + \frac{S_{pu}}{8} + 106 \\ &= S_{ov} + 105 \end{aligned} \tag{9}$$

where 32 bytes is the header length in the certificate.

5.2 Communication delay

The communication delay is defined as the time elapsed between the generation of a packet and its successful reception at the application layer. It includes the queuing delay and the medium service time (due to backoff, transmission delay, and propagation delay, etc.). Many delay analysis models for IEEE 802.11 MAC protocol have been proposed. To our knowledge, model from [31] is the best suited for VANET environment where there is no acknowledgement, and MAC layer retransmissions. In [31], the mean beacon transmission delay is defined as:

$$T_{tx} = \frac{W-1}{2}[\sigma P_e + T_S P_S + T_C P_C] + (1-\pi)^{n-1}(1-e)T_S \\ + (1-(1-\pi)^{n-1}(1-e))T_C \quad (10)$$

where W is the contention window, σ is the slot time, P_e, P_S, P_C are the probabilities of empty channel, successful transmission and collision respectively. T_S and T_C are the duration of successful transmission and collision respectively. These values depend on the packet size. π is the transmission probability in a slot by an active station, n is the total number of vehicles, e is the probability of a beacon packet corruption by noise, and T_h is the duration of both Preamble and PLCP header. From (10), we conclude that the communication delay depends on the packet size and the network density. Finally, the transmission delay is given by:

$$T_{tx} = \frac{W-1}{2}\left[\sigma P_e + \left(T_h + \frac{S_{ov}\times 8}{D_R} + DIFS + \delta\right)P_S \right. \\ \left. + \left(T_h + \frac{S_{ov}\times 8}{D_R} + EIFS + \delta\right)P_C\right] \\ + (1-\pi)^{n-1}(1-e)\left(T_h + \frac{S_{ov}\times 8}{D_R} + DIFS + \delta\right) \\ + \left(1-(1-\pi)^{n-1}(1-e)\right) \\ \times \left(T_h + \frac{S_{ov}\times 8}{D_R} + EIFS + \delta\right) \quad (11)$$

5.3 Simulation model and assumptions

All DSRC parameters used in this paper are listed in Table 4 and are IEEE 802.11p standard compliant. We conduct simulations using ns-2.34 within a 95% confidence interval. We make use of a Nakagami's probabilistic radio propagation model, because recent research has shown that a fading radio propagation model, such as the Nakagami's model is best suited for simulation of a WAVE environment [32, 33]. We use the ns-2 extensions provided by Chen et al. [34] as physical and MAC layer.

We consider the following scenario. In a highway of 5 km long, with 3 lanes in one direction, vehicles have a max velocity v_i where i is the lane number. Vehicle speeds are chosen according to speed limitation on French highway and average speed on a three lanes highway. We assume a uniform density β in veh/km/lane. Each node sends WSM of size S_A

Table 4 Simulation parameters

Parameter	Value
Propagation delay δ (μs)	1
Time slot σ (μs)	13
Packet size S_A (bytes)	73, 198, 256, 262
Vehicle density β (veh/km/lane)	[1; 45]
DIFS (μs)	64
EIFS (μs)	248
Packet interarrival time λ (s)	0.1
CWMin	15
Data rate D_R (Mbps)	6
Link Layer queue size (packets)	50
Vehicle speed (m/s)	$v_1 = 27.7$
	$v_2 = 30.5$
	$v_3 = 36.1$
Radio range R (meters)	300

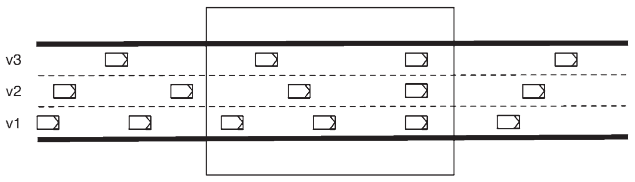


Fig. 4 Highway scenario

where A is the authentication chosen (WSM payload, WSM and certificate, WSM and certificate and P-224, WSM and certificate and P-256). According to safety-related applications requirements [5], each node generates one packet every 100 ms, and has a transmission range of 300 m for message exchange. We increase the density from 1 to 45 veh/km/lane, which means from free-flow to jam scenario. In our simulations, vehicles should enter the system in such a way that the network density remains stable. But in ns-2, all nodes are generated at the beginning of the simulation. Consequently, if the simulation has 600 nodes, then at $t = 0$ there are 600 nodes at the same place. As in standard ns-2, nodes could not be in sleep mode, they will participate to the network traffic even if they do not exist in reality. To avoid this undesirable effect, we monitor the traffic in an area $y = [2000; 3000]$, denoted by the box in Fig. 4.

5.4 Simulation results

There are six impacting parameters in a DSRC scenario: packet size, data rate, vehicle density, transmission power, message frequency and the number of lanes. The communication delay depends on the message size, the arrival rate at the MAC layer queuing system, the number of vehicles

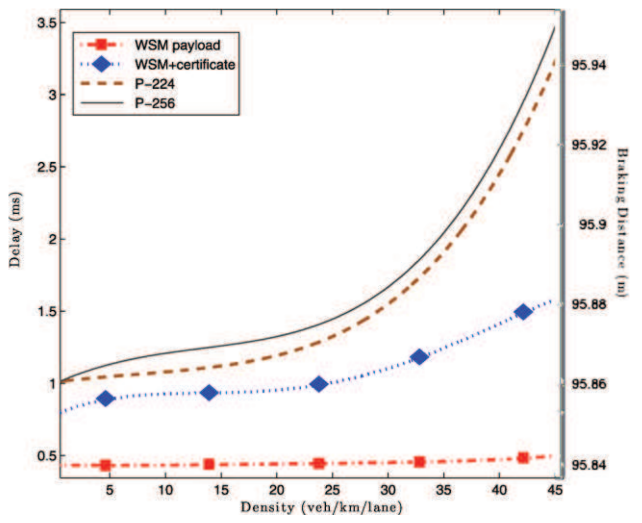


Fig. 5 Communication overhead: effect of density on delay and braking distance for different packet size

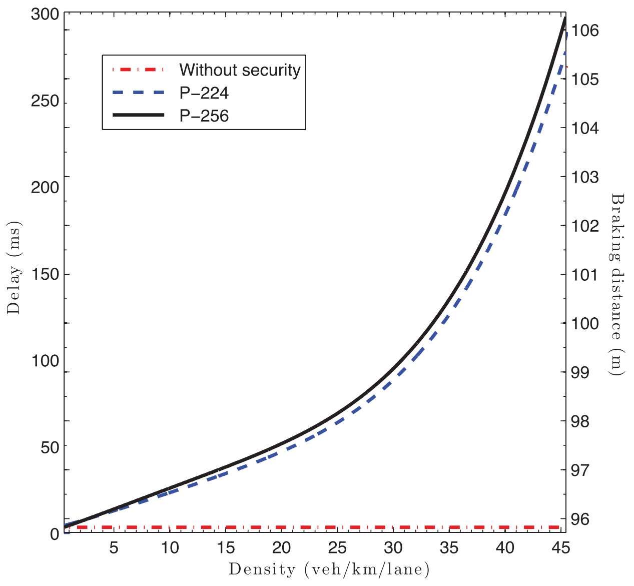


Fig. 6 Communication overhead: effect of density on delay and braking distance for *n* packets

within radio range, the probability of collision and the probability that the channel is busy. It is well established that larger the message size is, larger the probability of collision is. We focus on the effect of vehicle density because it affects the number of vehicles within radio range and the arrival rate at the MAC layer, and it increases the probability of collision or channel availability.

Figure 5 shows the impact of density on communication delay for one packet transmission. When the density increases, more vehicles are within the radio range. They compete for channel access, thus increasing the probability of collision. We simulated the same scenario with different packet size. WSM payload represents the WSM without security, i.e. 73 bytes. WSM + certificate represents the WSM appended with a signed certificate, i.e. 198 bytes. P-224 (respectively P-256) represents WSM + certificate signed with ECDSA and P-224 curve (resp. P-256 curve), i.e. 254 bytes (resp. 262 bytes). Figure 5 shows that without security, the density has a lower impact on the communication delay than in the other cases. According to the small size of WSM in this case, this result is obvious. Adding a security mechanism doubles the communication delay for density lower than 30 veh/km/lane. In high-density conditions, communication delay is multiplied by three. If we focus on the authentication key size, the comparison between P-224 and P-256 shows an overhead from 3% to 8%. The scale on the right of Fig. 5 shows the impact of density on braking distance. We notice that adding a security mechanism increases the braking distance from 2 cm to 10 cm for one packet. It may seem to be negligible but it is for one packet. So, if we are in the same scenario as in Sect. 4.2.2, a vehicle will have to check every signed messages received. The number of messages received depends on its neighborhood density. Figure 6 shows that the communication overhead adds until 10 meters to the normal braking distance (named "without security"). The worst case is when a vehicle needs to verify one packet per neighbor before making a decision. This case introduces the consensus problem which is detailed in Sect. 7.

6 Total time overhead of ECDSA

6.1 Definition

In Sects. 4 and 5, we investigate the processing overhead and the communication overhead. We merge both overheads to define the total time overhead of ECDSA according to formula (3).

$$
\begin{aligned}
T_{ov}(M) &= T_{sign}(M) + T_{tx}(Sign_{PrK_V}[M]) + T_{verify}(M)\\
&= (6n+2)T_{MUL} + T_{INV} + 5nT_{SQR} + T_{HASH}\\
&\quad + \frac{W-1}{2}\Bigg[\sigma P_e + \left(T_h + \frac{S_{ov}\times 8}{D_R} + DIFS + \delta\right)P_S\\
&\quad + \left(T_h + \frac{S_{pu} + 2\times(S_{sigcert}+S_{sigmess})+8}{D_R}\right.\\
&\quad \left. + EIFS + \delta\right)P_C\Bigg] + (1-\pi)^{n-1}(1-e)\\
&\quad \times \left(T_h + \frac{S_{pu} + 2\times(S_{sigcert}+S_{sigmess})+8}{D_R}\right.
\end{aligned}
$$

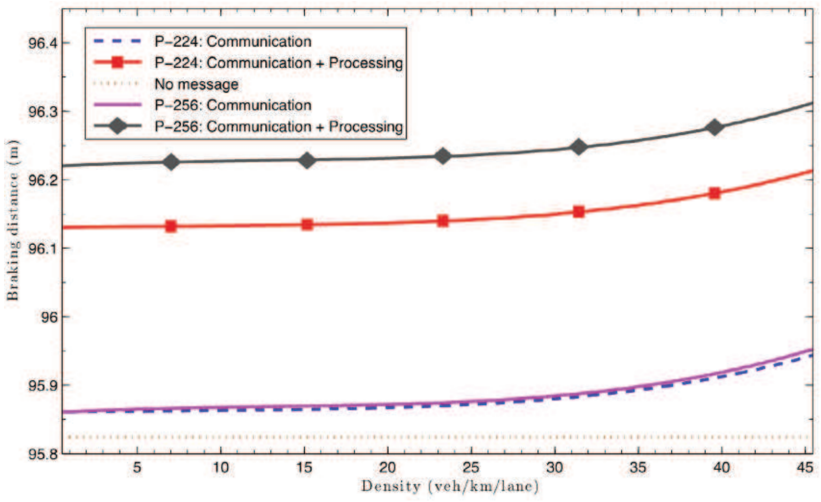


Fig. 7 Time overhead: effect of density on delay and braking distance for one packet

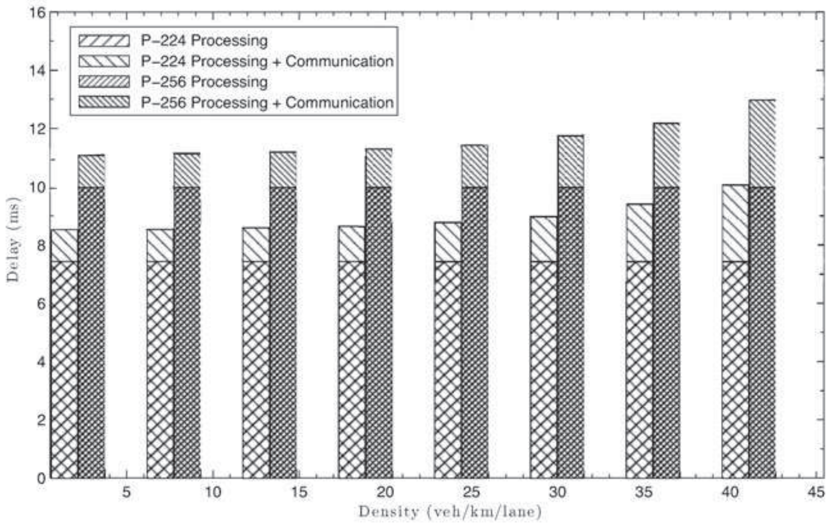


Fig. 8 Comparison between processing and communication overhead

$$+ DIFS + \delta\Big) + \left(1 - (1-\pi)^{n-1}(1-e)\right)$$
$$\times \left(T_h + \frac{S_{pu} + 2 \times (S_{sigcert} + S_{sigmess}) + 8}{D_R}\right.$$
$$\left. + EIFS + \delta\right) + (12n+2)T_{MUL} + T_{INV}$$
$$+ 10nT_{SQR} + T_{HASH} \qquad (12)$$

The formula (12) highlights that the total time overhead of ECDSA depends on the architecture of the OBU (utilization of crypto-processor or not), the authentication key size (P-224 or P-256), and the neighborhood density.

6.2 Impact on delay and braking distance

Figure 7 shows the processing and communication overhead of ECDSA. We observe that the processing mechanism adds 0.3 m to the braking distance for P-224 and 0.4 m for P-256. For one packet transmission, the processing overhead is greater than the communication overhead. Figure 8 confirms this result by highlighting the ratio of processing overhead into the total overhead. For one packet verification, more than 80% of ECDSA overhead is due to the processing mechanism.

As shown in Fig. 9, the processing delay, for n messages, is higher than the communication delay. For a density of 35 veh/km/lane, the communication delay is about 100 ms, while processing delay is 400 ms. Moreover, Fig. 9 details the difference between P-224 and P-256 delay. Using P-256 instead of P-224 has a greater impact on processing delay than on communication delay. Indeed, the communication curves are slightly different, while there is a gap between the two processing curves. Likewise, for a density of 35 veh/km/lane, communication adds 5 m to the braking distance, while the processing adds more than 17 m. One more time, using P-256 instead of P-224 has a greater impact on

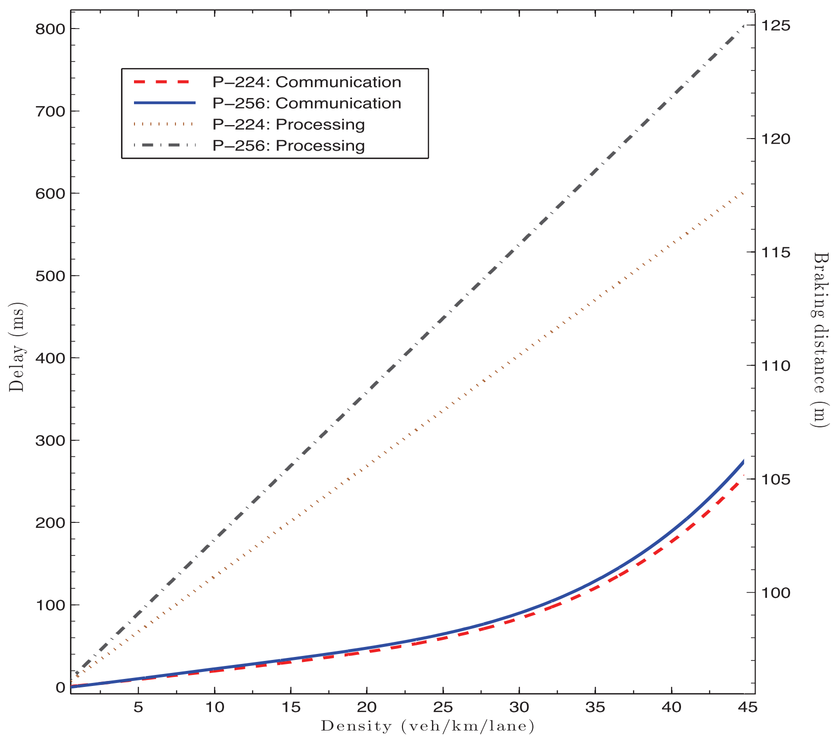


Fig. 9 Time overhead: effect of density on delay and braking distance for *n* packets

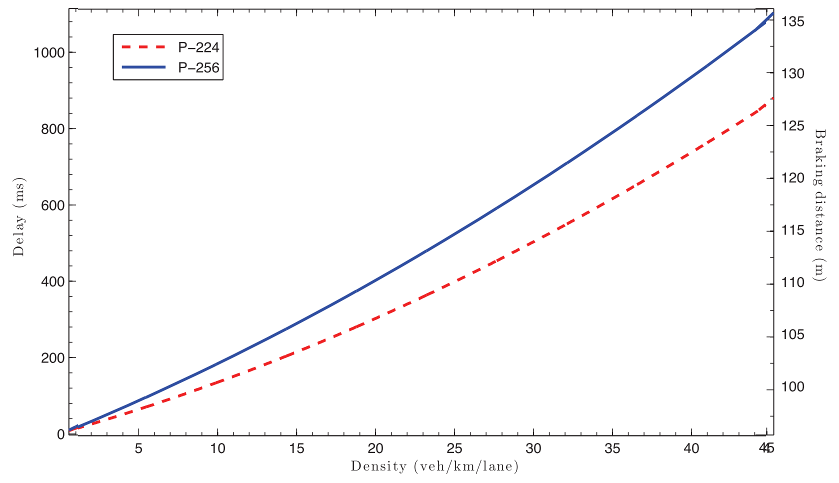


Fig. 10 Total time overhead: effect of density on delay and braking distance for *n* packets

the processing. Indeed, there is a gap of more than 5 m, which is greater than the average length of a car.

Figure 10 shows the total ECDSA overhead for one vehicle, which has to wait and check for *n* messages, depending on the density. In this figure, processing and communication delay are merged, and shows that the total overhead introduced by ECDSA is greater than 1 second for P-256 and 800 milliseconds for P-224 in high-density situations. The comparison between P-256 and P-224 shows an overhead of 30%. Moreover, in high-density scenarios, the braking distance is increased by more than 20 m. If P-256 is used, it adds an overhead from 1% to 8% higher than P-224. We conclude that the authentication has a significant impact on the delay and the braking distance. This security mechanism could have a high impact on the behavior of the application. Consequently, the authentication key size should be chosen carefully.

7 Consensus mechanism

7.1 Definition

Many applications depend on WAVE Short Message reception. For example, LDW application warns the driver in function of information included into the WSM. As pointed out before, security plays an important role in safety-related applications and V2V communications. For example, conventional solutions focus on securing the communications by utilizing digital signatures. But, since the detection of hazards is based on local sensor readings, an attacker may bilk the detection process of his vehicle. In this way, manipulating sensor readings to simulate a fake message may still result in a perfectly signed and certified message. Therefore, we evaluate the plausibility of information received during the decision process by a consensus mechanism. Thus, to avoid false information, application waits for x WSMs before warning the driver. This mechanism is called *consensus* [35]. The selection of x is an open issue. Four decision methods have been proposed in [8]. Our work is based on the decision method "*majority of Freshest X with Threshold*". According to [8], an important issue is the determination of the two parameters X and *Threshold*. This has to be based on the current traffic situation, which therefore has to be analyzed automatically by the LDW application. To set the parameter X, we propose and analyze four decision methods.

7.2 Decision methods

To support the decision process, we analyze four decision methods, which estimate the plausibility of a received warning by performing voting schemes. We detail how to set the consensus parameter x, which is the number of messages needed before making a decision.

We define two contrasting techniques:

- Static: x value never changed whatever the neighborhood density is.
- Dynamic: x is set in function of the current neighborhood density. As x is computed frequently, we denote it $x(t)$, where t is the current time of computation.

Moreover we define three dynamic methods. The *dynamic naive*, the *dynamic naive ahead*, and *majority ahead*.

(i) *Dynamic naive*: $x(t)$ is the number of one-hop neighbors at time t.
(ii) *Dynamic naive ahead*: $x(t)$ is the number of one-hop neighbors at time t ahead of the current vehicle.
(iii) *Majority ahead*: $x(t)$ is the half of the number of one-hop neighbors plus one at time t ahead of the current vehicle.

Dynamic decision methods compute the number of one-hop neighbors owing to WSMs received periodically.

Figure 11 shows the state transition diagram of a vehicle. A vehicle goes from *idle* to *sending alert* when it detects a hazard. The target of the hazard could be itself (the vehicle stops because of an emergency reason), another vehicle or the environment (ice, hole, obstacle). While the hazard is

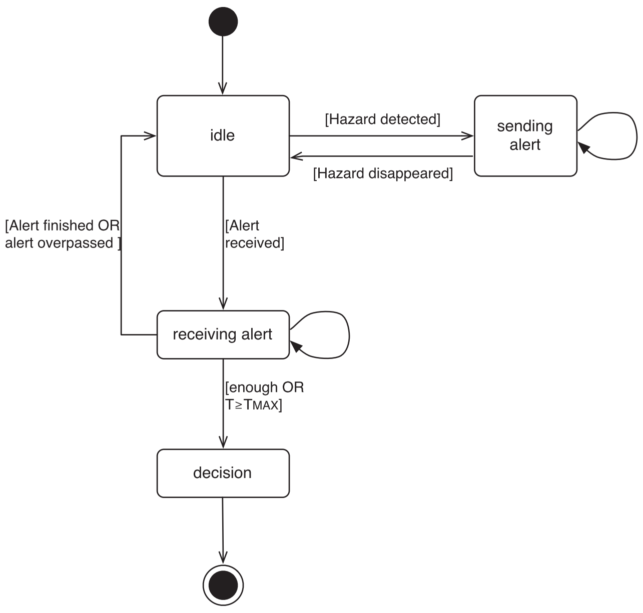


Fig. 11 State transition diagram of a vehicle

still detected, the vehicle will generate an alert. A vehicle goes from *idle* to *receiving alert* when it receives an alert. It keeps collecting WSMs until it reaches the threshold (i.e. the consensus parameter "enough"). It goes from *receiving alert* to *idle* when the hazard location is overpassed, or when the hazard has disappeared. The vehicle goes from *receiving alert* to *decision* when it receives enough WSMs (i.e. consensus parameter). Another case of transition could be the maximum delay allowed before making a decision. Indeed, safety-related applications have real-time constraints, and mandate to react before a specified delay (T_{MAX}). T_{MAX} is less than 500 ms for highly time-critical applications, and equal to three seconds for time-relevant applications [36]. In Fig. 11, T is the time between the first reception and the current time. In some cases, with the precautionary principle, a vehicle should make a decision even if it does not receive enough WSMs for this hazard.

7.3 Simulation model and assumptions

We assume that a certain fraction of the simulated vehicles will misbehave, which may result in fake attack. The goal of the fake attack is to trigger a false decision of the LDW application at the attacked vehicles. We define two simulation models.

(i) *Non-collaborative*: There is no collusion between attackers. Each attacker will generate its fake warning and broadcasts it. It will not help other attackers. Without collaboration, we assume that the probability of having multiple attackers reporting the same hazard is too low to consider this case.

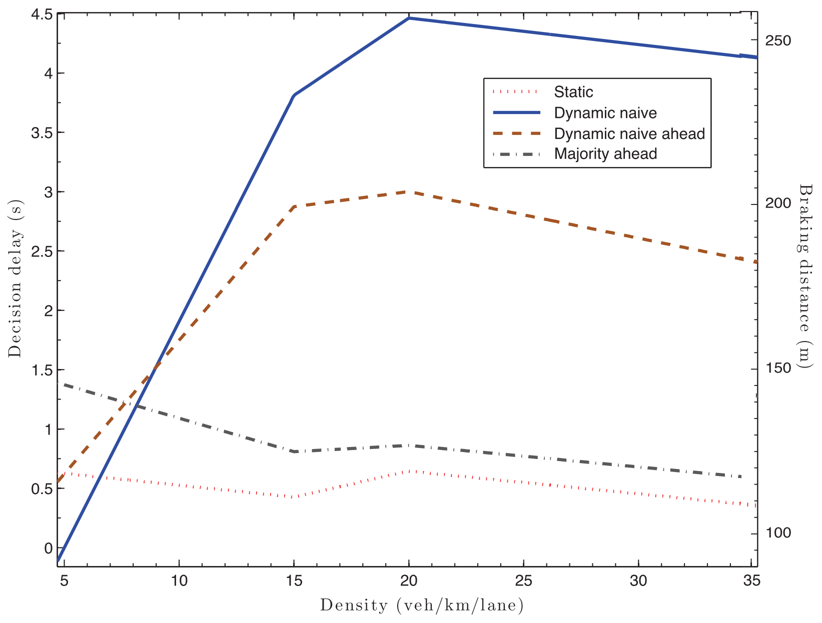


Fig. 12 Impact of consensus overhead on decision delay and braking distance

(ii) *Collaborative*: The attackers collude to make the fake warning faster accepted. In other words, when an attacker receives a fake warning, it will generate a fake warning concerning the sender.

We assume that a node cannot possess multiple identities, and has a majority of honest one-hop neighbors. We do not consider a forwarding mechanism (i.e., a vehicle generates a warning only when it detects a hazard). The results presented are from a scenario with 10% of attackers and 10% of honest vehicles will generate a warning. We set $x = 2$ for the *static* method because in simulation context (i), two messages are enough to avoid fake attacks. For the *dynamic* methods, we define $x = \min(x_{MAX}, DYN)$. DYN is the result of the *dynamic* method (cf. Sect. 7.2), and x_{MAX} is the maximum number of messages needed depending on T_{MAX} value.

7.4 Simulation results

Figure 12 shows the decision delay for the four decision methods analyzed. The decision delay is the time between the reception of the first warning and the decision. As P-224 and P-256 shares the same performance, we only present P-224 results. The *static*, *majority ahead*, and *dynamic naive all* methods show a percentage of false decision equal to zero. This could be explained because to reach a false decision, a vehicle needs to receive twice the fake warning from two different vehicles. But, in the current scenario (i.e. scenario (i)), there is no collaboration between attackers. So, a vehicle will always receive only one fake warning. That is the reason why the *static* method is the best in this context. Only two warnings will be enough to avoid fake attack. In the context of simulation (ii) (with collaboration and honest majority), the best method is the *mean*. Indeed, in a context of honest majority, $\frac{n(t)}{2}$ is enough to avoid fake attack ($n(t)$ is the current number of neighbors at time t).

Figure 12 shows that *static*, *majority ahead*, and *dynamic naive ahead* respect the maximum delay T_{MAX} (3 seconds here). Moreover, the *mean* method has the lowest decision delay because of the lowest consensus parameter (i.e. $x = 2$). On the scale on the right of the figure, we remark that the *static* and *mean* methods have the lowest overhead of the braking distance. Indeed, the maximum overhead is 20 meters. Moreover, the braking distance overhead is stable when the density increases. From 15 veh/km/lane, the two other methods doubles the braking distance. This underlines the need of a forwarding mechanism to increase the dissemination area. Thus, this overhead will be less critical. We also notice that from 15 veh/km/lane the overhead is stable. That is due to the threshold set to T_{MAX}.

Figure 13 shows the impact of consensus and ECDSA on the braking distance. We add T_{ov} of (3) to the decision delay, and translate it on braking distance. Once again, *static*, and *majority ahead* have the lowest braking distance overhead and it remains stable. Indeed, the braking distance overhead is greater than 25 meters. In high-density scenarios, *dynamic naive* and *dynamic naive ahead* methods multiply by 7 the braking distance.

8 Discussion

In Sects. 4 and 5, we investigate the authentication overhead of ECDSA. We conclude that the processing time overhead is of paramount importance in the total time overhead of

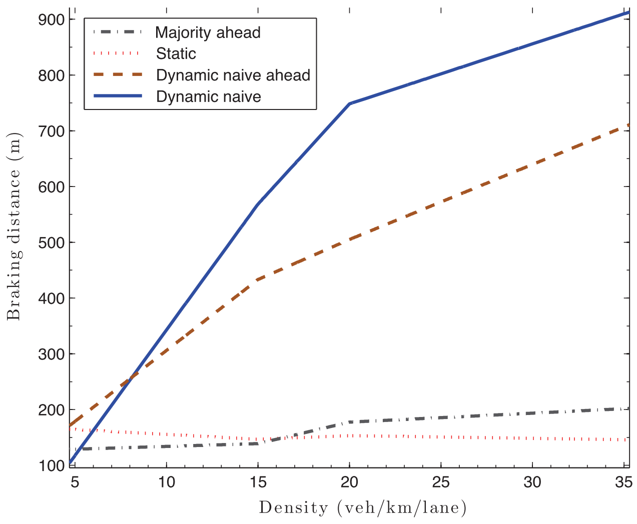


Fig. 13 Impact of consensus and ECDSA on braking distance

ECDSA. To improve the computational overhead, a crypto-processor dedicated to elliptic curve cryptography could be interesting. Moreover, in [37], authors proposed to not attach certificates to all messages, but rather for one every α successive beacons; they also proposed certificate caching to reduce verification processing overhead and the packet size overhead. In [38], authors proposed to dynamically change the transmission power depending on the vehicle density to reduce the N_{TX}.

Section 7 introduces the problem of consensus. It shows that the consensus avoids fake attacks. We notice that the context influences the decision method used in the consensus mechanism. We suggest a technique to switch between methods in function of the current context.

9 Conclusion and future work

VANETs deployment has the potential to greatly increase vehicular safety and improve driving experience. But, vehicular communications need to be secured. Therefore, the DSRC standard for vehicular ad hoc networks is based on the ECDSA algorithm for supporting authentication mechanism. But, security mechanisms come with overheads that affect the performance of the V2V communications, and hence that of the safety applications. In this paper, we investigate the total overhead of ECDSA, combining the packet size, processing and communication overheads. We focus on safety applications, and analyze the impact of the authentication on the braking distance. We conduct simulation study in order to evaluate the performance of secured beacon safety message dissemination in vehicular ad hoc networks. We pay special attention to safety requirements while studying networking performance issues.

Our results show that the processing overhead is higher than the communication overhead. Depending on the application requirements, the braking distance is increased by more than an average length of a car in high-density scenario. We highlight the impact of the authentication key size in order to adapt security parameters to the application requirements. Some optimizations were proposed.

To avoid false data dissemination, we introduce the problem of consensus. We analyze four decision methods to check the data consistency. The *static* method shows the best results in non-collaborative scenario. In collaborative scenario, the dynamic method permits to change the number of messages needed in function of density. Moreover, in the context of a majority of honest vehicles, the *mean* method shows the best results.

As of future work, we intend to enhance the authentication overhead assessment by adding the certificate distribution, verification and revocation mechanisms. Indeed, when a node receives a WAVE short message, it has to check the certificate appended to the message. The IEEE 1609 standards mandate the use of CRL, but do not specify how to deal with. Moreover, we will complete our work on consensus by adding an indicator to further improve the consensus overhead and the performance of the decision process.

References

1. European Road Safety Observatory (2007). *Traffic safety basic facts 2007* (Technical Report).
2. Blincoe, L. (2002). *The economic impact of motor vehicle crashes* (Technical Report). U.S. Department of Transportation National Highway Traffic Safety Administration.
3. NHTSA (2008). Traffic safety fact sheet.
4. CARE (2007). Community road accident database.
5. NHTSA (2006). Vehicle safety communications project—FINAL REPORT VSC. CAMP IVI Light Vehicle Enabling Research Program, DOT HS 810 591.
6. Miller, V. S. (1985). Use of elliptic curves in cryptography. In *LNCS: Vol. 218. Advances in cryptology (CRYPTO)* (pp. 417–426).
7. National Institute of Standards and Technology (1999). Recommended Elliptic Curves for Federal Government Use. NIST.
8. Ostermaier, B., Dötzer, F., & Strassberger, M. (2007). Enhancing the security of local danger warnings in VANETs—a simulative analysis of voting schemes. In *2nd international conference on availability, reliability and security (ARES)* (pp. 422–431).
9. Iyer, A., Kherani, A., Rao, A., & Karnik, A. (2008). Secure V2V communications: Performance impact of computational overheads. In *IEEE conference on computer communications workshops (INFOCOM)* (pp. 1–6).
10. Haas, J. J., Hu, Y., & Laberteaux, K. P. (2009). Real-world VANET security protocol performance. In *IEEE globecom symposium on selected areas in communications*.
11. Rao, A. (2009). *Performance evaluation of secure communication in vehicular networks*. Master Thesis, Indian Institute of Technology Delhi, January 2009.
12. IEEE (2006). *IEEE Trial-use standard for wireless access in vehicular environments—security services for applications and management messages*. IEEE Standard 1609.2-2006.

13. ANSI. Public key cryptography for the financial services industry: the elliptic curve digital signature algorithm. ANSI X9.62-1998.
14. Koblitz, N. (1987). Elliptic Curve Cryptosystems. *Mathematics of Computation*, *48*, 203–209.
15. Certicom (2000). Standards for efficient cryptography, SEC2: recommended elliptic curve parameters.
16. Johnson, D., Menezes, A., & Vanstone, S. (2001). The elliptic curve digital signature algorithm (ECDSA). *International Journal of Informatics Security*, *1*(1), 36–63.
17. Negre, C. (2005). *Scalar Multiplication on Elliptic Curves Defined over Fields of Small Odd Characteristic*. *Lecture Notes in Computer Science* (Vol. *3797*, pp. 389–402).
18. Järvinen, K., & Skyttä, J. (2007). Final project report: cryptoprocessor for elliptic curve digital signature algorithm (ECDSA).
19. Okeya, K., & Sakura, K. (2001). Efficient elliptic curve cryptosystems from a scalar multiplication algorithm with recovery of the *y*-coordinate on a Montgomery-form elliptic curve. In *3th international workshop on cryptographic hardware and embedded systems* (pp. 126–141).
20. Guan, D. J. (2003). Montgomery algorithm for modular multiplication. Available: http://isl.cse.nsysu.edu.tw/note/montg.pdf.
21. Menezes, A. J., Van Oorschot, P. C., & Vanstone, S. A. (2001). *Handbook of applied cryptography*. Boca Raton: CRC Press.
22. Kornerup, P. (1993). High-radix modular multiplication for cryptosystems. In *11th symposium on computer arithmetic* (pp. 277–283).
23. Kaihara, M. E., & Naofumi Takagi, N. (2005). A hardware algorithm for modular multiplication/division based on the extended Euclidean algorithm. In *IEICE transactions on fundamentals of electronics, communications and computer sciences* (Vol. E88-A, pp. 3610–3617).
24. Ma, S., Hao, Y., Pan, Z., & Chen, H. (2008). Fast implementation for modular inversion and scalar multiplication in the elliptic curve cryptography. In *2nd international symposium on intelligent information technology application* (Vol. 2, pp. 488–492).
25. Lopez, J., & Dahab, R. (1999). Fast multiplication on elliptic curves over GF(2m) without precomputation. *Cryptographic Hardware and Embedded Systems*, *1717*, 316–327.
26. National Institute of Standards and Technology (2002). Fips 180-2, secure hash standard, federal information processing standard (fips). Publication 180-2.
27. Raya, M., & Hubaux, J. P. (2005). The security of vehicular ad hoc networks. In *3rd ACM workshop on security of ad hoc and sensor networks* (pp. 11–21).
28. Abusharekh, A., & Gaj, K. (2007). Comparative analysis of software libraries for public key cryptography. In *ECRYPT workshop on software performance enhancement for encryption and decryption* (pp. 1–16).
29. Wischhof, L. (2007). *Self-organizing communication in vehicular ad hoc networks*. Ph.D. thesis, Hamburg-Harburg University.
30. IEEE Vehicular Technology Society. 5.9 GHz dedicated short range communications (DSRC)—overview.
31. Vinel, A., Andreev, S., Koucheryavy, Y., & Staehle, D. (2009). Estimation of a successful beacon reception probability in vehicular ad-hoc networks. In *ACM international conference on wireless communications and mobile computing: connecting the world wirelessly (IWCMC)* (pp. 416–420).
32. Schmidt-Eisenlohr, F., Torrent-Moreno, M., Mittag, J., & Hartenstein, H. (2007). Simulation platform for inter-vehicle communications and analysis of periodic information exchange. In *4th conference on wireless on demand network systems and services (WONS)* (pp. 50–58).
33. Taliwal, V., Jiang, D., Mangold, H., Chen, C., & Sengupta, R. (2004). Empirical determination of channel characteristics for DSRC vehicle-to-vehicle communications. In *1st ACM international workshop on vehicular ad hoc networks (VANET)* (p. 88).
34. Chen, Q., Schmidt-Eisenlohr, F., Jiang, D., Torrent-Moreno, M., Delgrossi, L., & Hartenstein, H. (2007). Overhaul of IEEE 802.11 modeling and simulation in NS-2. In *10th ACM symposium on modeling, analysis, and simulation of wireless and mobile systems (MSWiM)* (pp. 159–168).
35. Cao, Z., Kong, J., Lee, U., Gerla, M., & Chen, Z. (2008). Proof-of-relevance: filtering false data via authentic consensus in vehicle ad-hoc networks. In *IEEE conference on computer communications workshop (INFOCOM)* (pp. 1–6).
36. Kargl, F., Ma, Z., & Schoch, E. (2006). Security engineering for VANETs. In *4th workshop on embedded security in cars (ESCAR'06)*
37. Calandriello, G., et al. (2007). Efficient and robust pseudonymous authentication in VANET. In *ACM VANET* (pp. 19–28).
38. Elbatt, T., Goel, S. K., Holland, G., Krishnan, H., & Parikh, J. (2006). Cooperative collision warning using dedicated short range wireless communications. In *ACM VANET* (pp. 1–9).

Jonathan Petit is a Ph.D. candidate at Paul Sabatier University (University of Toulouse). He received the M.S. degree in Computer and Communications Engineering in 2007 from the Paul Sabatier University, Toulouse, France. His research interests include security in wireless ad hoc networks, vehicular networks, quality of service and performance evaluation.

Zoubir Mammeri received his M.S., Ph.D. and Habilitation in computer science from National Polytechnic Institute of Lorraine (France) in 1982, 1985 and 1995. Since 1998, he is a full professor at Paul Sabatier University at Toulouse. His research interests include: quality of service, QoS-based routing, packet scheduling, mobile ad hoc networks, sensor networks, security in sensor and vehicular networks, real-time systems, scheduling, real-time distributed systems. In the last fifteen years, he served as program committee member of over one hundred international conferences and workshops. He published several papers and books on real-time systems and communication networks. He is a senior member of IEEE and IFIP member.