Real-World VANET Security Protocol Performance

Jason J. Haas and Yih-Chun Hu
University of Illinois at Urbana-Champaign
Urbana, Illinois, U.S.A.
{jjhaas2,yihchun}@illinois.edu

Kenneth P. Laberteaux
Toyota Research Institute
Ann Arbor, Michigan, U.S.A.
klaberte@acm.org

***Abstract*—Many results have been published in the literature based on performance measurements obtained from simulations of Vehicular Networks (VANETs). These simulations use as input traces of vehicle movements that have been generated by traffic simulators which are based on traffic theory models. To our knowledge, no one has published any work based on actual large-scale recordings of vehicle movements. We use recordings of actual vehicle movements on various roadways. In order to enable analysis on this scale, we have developed a new VANET simulator, which can handle many more vehicles than NS-2 [1]. To enable us to use our own simulator, we present results of a cross-validation between NS-2 and our simulator, showing that both simulators produce results that are statistically the same. We use our simulator to analyze the proposed authentication mechanism, which relies on ECDSA signatures [2], comparing it to broadcast authentication using TESLA [3]. We perform our evaluations using real vehicle mobility, which we believe to be the first simulations using real vehicle mobility. Our comparison shows strengths and weaknesses for each of these authentication schemes in terms of the resulting reception rates and latency of broadcast packets.**

I. Introduction

There are many envisioned applications for VANETs: vehicle safety enhancement, traffic congestion notification, emergency notification, electronic tolling, ad dissemination, and media download. In the U.S., the FCC has delegated 75 MHz for DSRC (VANET radios) use in the 5.9 GHz band. Similarly, in Europe, the EU has dedicated 30 MHz to vehicle-to-vehicle communication. Standards are being assembled for DSRC PHY and MAC layers in IEEE 802.11p. The IEEE 1609 set of standards specifies upper layer operation. Safety messages are likely to be sent at a rate up to 10 Hz [4]. There are two modes of vehicle communication in a VANET: vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I).

Safety messages (i.e., heartbeats) are a form of V2V communication that contain vehicle state information such as position, velocity, acceleration, and brake status. These messages can be used to warn drivers of vehicles that may have otherwise been unseen without these warnings. The hope is that these warnings will improve automotive safety, reduce the loss of life in vehicle-related crashes, and save large amounts of money from reduced healthcare costs.

Previous work has focused on simulating VANET environments using traffic theory or traffic generation tools based on traffic theory. These simulations have been used to analyze network behavior and are based on parameters such as node degree, vehicle speed, and vehicle density (e.g. vehicles/lane/km) [5]. To our knowledge, no one has yet simulated either VANET safety message network-wide performance using actual measured vehicle traces. In this paper, we present simulations based on *real vehicle movements* [6].

In order to simulate the real vehicle movements we obtained, we built our own custom DSRC simulator. To verify results obtained from our simulator, we cross-validated results from our simulator with results from NS-2 using NS-2's DSRC extensions [1].

Additionally, we analyze the end-to-end performance of broadcast safety messages. We simulate the network performance of safety messages using ECDSA signatures, as specified in the draft IEEE 1609.2 standard [2]. We also simulate safety message performance while using TESLA-based authentication [3]. Finally, we simulate the use of different processors for verifying signatures, showing the latency of messages for various hardware. Investigating channel congestion caused by safety messages is important because the results show how much of the channel is required for safety messages, being the most important source of packets in VANETs, and how much of the channel remains for other ad hoc services[1].

The remainder of this paper is organized as follows. In Section II, we cover previous work. We present our cross-validation of simulators in Section III. In Section IV, we show the performance at various levels for safety broadcasts. Finally, we conclude in Section V.

II. Related Work

Yin et. al [8] review DSRC and safety applications in VANET. The authors briefly review the PHY layer of the DSRC environment, present applications for safety broadcasts, and argue that safety messages are the priority messages for VANETs since the primary goal of deploying VANETs is to save lives. The authors also provide results from a detailed PHY-level network simulation of safety broadcast messages using the Qualnet network simulator, showing the message reception percentage for varying frequencies of safety message broadcasts and the latency associated with those frequencies. The simulations were based on traffic traces generated by CORSIM on a map of city streets and used 100 vehicles in an area of 6,600x4,200 meters. The authors used a transmission

[1]Knowing these results will let designers specify the channel switching duty cycle for IEEE 1609.4 [7]

978-1-4244-4148-8/09/$25.00 ©2009

power of 17 dBm, which resulted in a transmission range of approximately 300 meters.

Robinson et. al, [9] present a framework for using and generating safety broadcast messages. The authors design an architecture for on-board units (OBUs) and implement their architecture in two vehicles. They also give various uses for safety broadcast messages, such as, traffic signal violations warnings, curve speed warnings, emergency braking warnings, pre-crash warnings, and lane change warnings. The frequencies required for these applications range from 1-50 Hz. The transmission range required for these applications ranges from 50-300 meters.

Hu and Laberteaux [3] consider various design issues with safety messages. The authors compare ECDSA/PKI-based authentication, to broadcast authentication using TESLA. The authors provide a brief review of TESLA. They argue that a VANET could reduce network congestion by using TESLA because of TESLA's reduced packet size and reduce the computational overhead of signature verification since TESLA uses cryptographic hash functions to generate signatures.

Studer et. al [10] proposed using a modified version of TESLA called TESLA++ with ECDSA signatures together called VAST. This hybrid use of signatures provides resilience to DoS attacks on both memory and processor usage, and provides non-repudiation. Using either TESLA or ECDSA alone do not provide all of these properties. The authors simulated TESLA, ECDSA, and VAST authentication using NS2 on a 1 km long highway, varying the number of vehicles within radio range between 1 and 75 vehicles, using a radio range of 300 m. Their simulation of TESLA and VAST sends the two required pieces of information (heartbeat and key for TESLA, and MAC and heartbeat with key for VAST) in a single packet, attaching the key for TESLA or the message and key for VAST to the next packet which contains the safety message for TESLA or the MAC for VAST, thus reducing the number of packets sent and correspondingly network congestion. Thus, there is only a single packet sent for each safety message. The authors' simulations use a traffic density of approximately 15.6 vehicles/km/lane in a highway environment, which is of much lower density than the real recordings we will use and discuss below.

III. Cross-Validation

A. Motivation

The NGSIM project [6], supported by the United States Federal Highway Administration, has recorded vehicle movements on various roadways in the United States. These roadways include I-80 in Emeryville, California, Lankershim Boulevard and US-101 in Universal City, California, and Peachtree Street in Atlanta, Georgia. Lankershim and Peachtree are major arterial roads, and I-80 and US-101 are major multi-lane highways. These data sets record the movements of thousands of vehicles. For our comparisons below, we chose the I-80 and Lankershim traces, since I-80 held the densest (most congested) traffic and Lankershim since it was an arterial roadway, not a major highway. We chose to use the Lankershim data over the Peachtree data because the Peachtree data was released after we had already completed a significant number of simulations using the Lankershim data. NGSIM obtained vehicle movements from transcribing video of the simulation area at a rate of 10 frames per second, i.e., vehicle positions are updated every 0.1 seconds. For each roadway, two or three traces were recorded, each being approximately 15 minutes long.

The I-80 data was gathered over a section of the interstate that is mostly straight and includes an on-ramp. There are three sets of data available from NGSIM for I-80: 4:00-4:15pm, 5:00-5:15pm, and 5:15-5:30pm. These times are during rush hour. We chose to perform our cross validation using the 4:00-4:15pm (I-80 4:00pm) data.

Traces obtained from the NGSIM project contain up to 2,169 vehicles and last for approximately 15 minutes of simulated time. The I-80 and US-101 traces, as obtained from the project, consisted of only a single direction of traffic. Simulating these traces directly would not have given realistic worst-case results. Thus, we geometrically reflected the traffic and shifted it such that the reflected traffic moves in the opposite direction and is in the place the traffic from the opposite direction would be. This resulted in a doubling of the number of simulated vehicles (up to 4,338). We will describe the real vehicle movement traces we used in other simulations in greater detail in Section IV-B below.

a) Advantages Over NS-2: We built extensions to NS-2 to simulate safety messages and estimated that the real-world duration of simulating traces of this size would be at least 100 days given current computer hardware. Thus, we built our own custom packet-based, event-driven simulator, which can simulate this amount of traffic data approximately 600 times faster, that is, in a matter of a few hours. Our simulator is custom built for VANET simulations and thus is a specialized tool. Being a specialized tool, our simulator does not include all of the other overhead of the supporting code for other applications. For example, the NS-2 VANET extensions [1] contain a more complete version of the 802.11p MAC, while our simulator currently supports only broadcast. Additionally, we have profiled the simulator code to optimize it for speed. Specifically, we have noted that the in-order handling of simulation events is a major consumer of CPU time. As a result, we chose to use a heap scheduler rather than a linked list for the global event queue and have attempted to reduce the number of events generated.

B. Validation Methodology

In order to enable us to use our own custom simulator to study large-scale VANETs, we validated results from our simulator with those obtained from NS-2 [1]. In each simulator, we built extensions for broadcasting safety messages from each vehicle every 0.1 seconds. These safety messages were signed using ECDSA signatures, as specified by the IEEE 1609.2 draft standard. We used the deterministic two-ray ground channel fading model for all comparisons in this

978-1-4244-4148-8/09/$25.00 ©2009
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2009 proceedings.

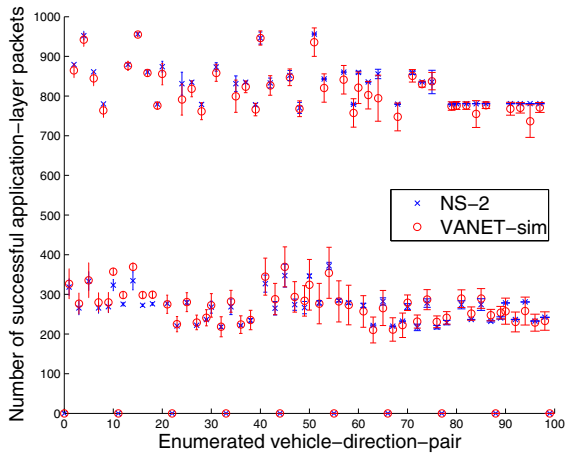


Fig. 1. A 10-vehicle subset of our cross-comparison validation results

section. All safety messages were 142 bytes long, including MAC and PHY layer headers.

For our cross comparisons, we used the first 50 vehicles that enter the I-80 4:00pm simulation area as our test data set. We compared the number of packets for each vehicle-destination-pair between NS-2 and our simulator. A vehicle-destination-pair consists of two different vehicles in the simulation, e.g., the i-th vehicle and the j-th vehicle to enter the simulation area. We consider each direction of message transmission separately, that is, messages broadcast from the i-th vehicle and received by the j-th vehicle, and messages broadcast from the j-th vehicle and received by the i-th vehicle are considered separately.

We chose to cross-compare successful application layer packets because we will use a measure of these packets in one of our performance analysis for our simulations presented below. Since one of our goals is to determine how a VANET might perform in a real deployment scenario, the number of successful application layer safety messages is what will determine the utility of the VANET in vehicle safety applications.

We performed 10 simulations of the same test data set (the first 50 vehicles to enter the I-80 4:00pm data set) both in NS-2 and in our simulator. For each vehicle-direction-pair, we computed the average number of messages received at the application layer. We also computed the 95% confidence intervals for each vehicle-direction pair average over those 10 simulations.

C. Validation Results

A subset of 10 vehicles of the resulting comparison is shown in Figure 1. We omit the full 50-vehicle comparison because the resulting plot is unreadable in this format. The distribution of vehicle-destination-pairs is bimodal because the traffic during this time-period is formed into two groups. Each group enters the simulation area at opposite ends of the roadway, one from the top the other from the bottom. The section of I-80 that is recorded in this data set is approximately 503 meters long. Thus two vehicles entering the simulation area from opposite ends of the roadway will have a much lower chance of successfully receiving a packet than vehicles entering the roadway from the same end of the roadway. The pairs that enter at the same end lie in the higher success mode, and the pairs that enter at opposite ends lie in the lower success mode. The pairs that have no successful receptions are same-vehicle pairs, that is, vehicles never receive packets from themselves since they are transmitting when they would need to be receiving from themselves.

This graph shows that almost every vehicle-direction-pair is within error bars across simulators. Thus, any simulation that would be performed using NS-2 can be performed on our simulator with the same confidence in the results.

We also computed the mean of the absolute value of the percent difference and mean percent difference in packets successfully received at the application layer across simulators for all vehicle-direction pairs. The result of this calculation was a 3.53% mean absolute percent difference and -1.52% mean percent difference between simulators. The latter number indicates that results from our simulator show 1.52% more packets being received at the application layer on average. Thus, even if the statistical similarity demonstrated by the error bars is questioned, our results should on average only need to be adjusted by subtracting 1.52% from our application layer success rate.

IV. Safety Message Performance Analysis

In this section, we will investigate the performance trade-offs between using ECDSA signatures as proposed in the IEEE 1609.2 draft standard [2] and using TESLA signatures, as proposed by Hu and Laberteaux [3]. We perform an analysis comparing ECDSA and TESLA authentication to determine which scheme results in less network congestion. Specifically, we will investigate application layer reception rate. We will also compare the computational overhead of verifying signatures.

IEEE 1609.2 [2] specifies using ECDSA signatures over cryptographic hashes using either SHA-224 or SHA-256. The curves used for ECDSA are specified as either nistp224 or nistp256, which result in signatures 56 or 64 bytes long, respectively (two numbers are required to make a signature). In our simulations, we chose to use nistp224 and SHA-224, resulting in the fastest and smallest signatures.

TESLA [11] uses time as the mechanism for creating asymmetric knowledge, relying on predictable releases of keys for security. Since accurate timing is required by TESLA, vehicles using TESLA signatures will require a clock source for synchronizing their local clocks. This clock source is conveniently already available in the GPS hardware that vehicles will make use of for determining their positions. TESLA signatures are message authentication codes, which are generated using cryptographic hash functions (e.g., SHA-1). Using hash functions to generate signatures results in orders of magnitude smaller verification times as compared with using ECDSA to generate signatures. Additionally, using

978-1-4244-4148-8/09/$25.00 ©2009
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2009 proceedings.

hash functions for signatures instead of ECDSA signatures significantly reduces the packet overhead due to signature size. For our TESLA simulations, we used 10-byte long signature fields and HMAC-SHA-1 for generating signatures [3]. Further details of TESLA can be found in earlier work [3; 11].

A. Authentication Mechanism Simulation Methodology

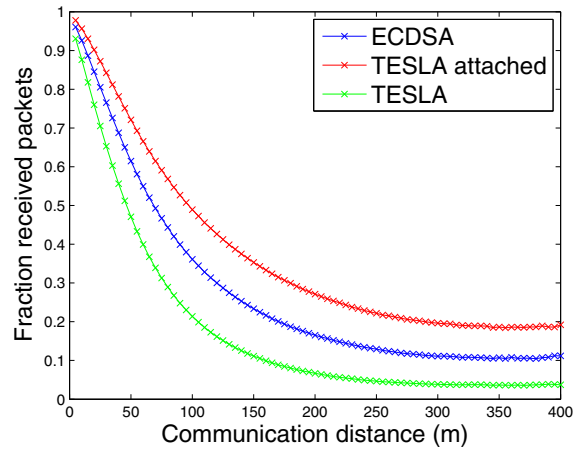
In addition to ECDSA-based authentication, as specified in the IEEE 1609.2 draft standard, we implemented TESLA-based authentication in our simulator. When a TESLA key was used to create a signature, the key was either released in a separate packet after the safety message broadcast or attached to the following broadcast. This latter behavior was implemented to reduce the network load induced by separate key packets and we will refer to this behavior as "TESLA attached" below. Including all header overhead (e.g., PHY-layer and MAC-layer headers), safety messages with TESLA authenticators were 82 bytes long and with the attached keys scheme were 92 bytes long. Key packets, when keys were not attached to position packets, were 43 bytes long. Each vehicle broadcast its safety message every 0.1 seconds.

Attaching keys to safety message broadcasts reduces the load on the network by reducing the number of packets broadcast. However, delaying keys until the following safety message packet results in increased latency for verification because vehicles cannot verify a signature until they receive the key in the following safety message broadcast from the sending vehicle, which is sent approximately 0.1 seconds later. We did not simulate certificates being distributed because certificate distribution is a separate issue. Because TESLA uses a hash chain to generate its keys, we allowed vehicles to use any following key to verify authenticators, thus a packet's signature can be verified by not just the intended next key but also by keys released later. Allowing vehicles to use these latter keys results in many more additional verifications being possible, as our simulations will show below.

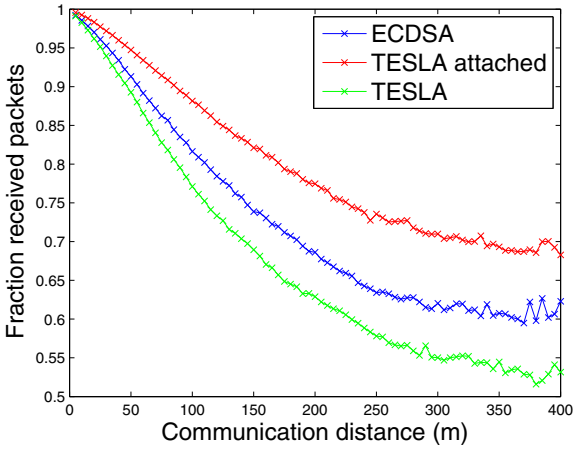
B. Real Vehicle Movements

We chose to use the I-80 5:00pm and Lankershim 8:45am data sets for our comparisons of security protocols. Here we describe the real vehicle traces we used in greater detail.

The I-80 data was gathered from a section of that interstate that is approximately 503 meters long. This section of roadway contains 6-7 lanes of traffic and an on-ramp. Of the 3 data sets for I-80, we chose the 5:00pm data set because it contains the most dense traffic, having 1,836 vehicles. As described above, this data was from a single direction of traffic only, and we reflected and offset the traffic, adding this to the original traffic to produce traffic that mimicked bidirectional traffic on a highway. The result was a trace consisting of 3,672 vehicles. Not all of these vehicles are in the simulation area concurrently. The simulation time is approximately 15 minutes. The average density of this trace is 52.1 vehicles/km/lane. The average speed of vehicles in this recording is 20.3 km/h. The traffic in the left-most lane in each direction are high-occupancy vehicle (HOV) lanes and moves considerably faster than the other lanes of traffic.



(a) I-80



(b) Lankershim

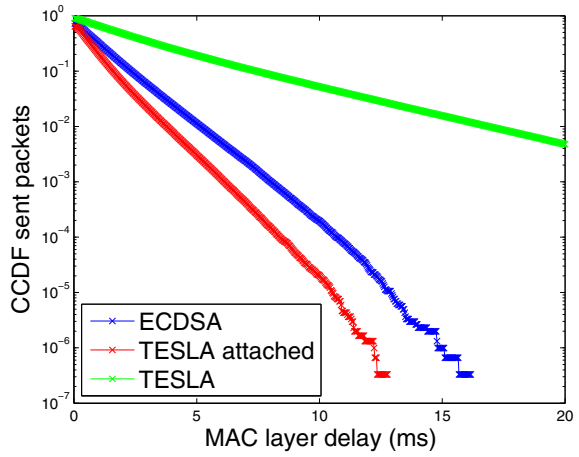
Fig. 2. Network layer reception performance of ECDSA-based and TESLA-based authentication

The Lankershim Boulevard data sets were gathered from a section of the road that was approximately 488 meters long. There are 3-4 lanes in each direction in this area. The recorded area contains 4 intersections controlled by traffic signals. Traffic was recorded for both directions along Lankershim and also for a small section of the cross traffic at the intersections. We chose to simulate the 8:45am traffic on this roadway, being the denser of the two Lankershim traces. This trace contains 1,231 vehicles. Again, not all of these vehicles are in the simulation area concurrently. This trace lasts approximately 15 minutes of simulated time. The average density of this trace is 28.7 vehicles/km/lane, and the average speed of vehicles is 18.1 km/h.

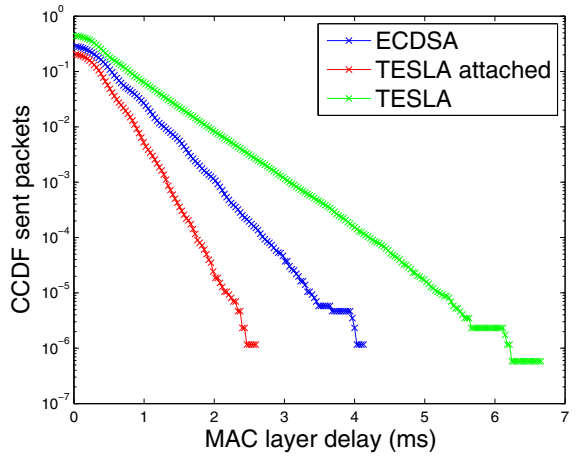
C. Network Reception

Figures 2(a) and 2(b) show the fraction of packets received from the network layer versus distance for the I-80 and

978-1-4244-4148-8/09/$25.00 ©2009



(a) I-80



(b) Lankershim

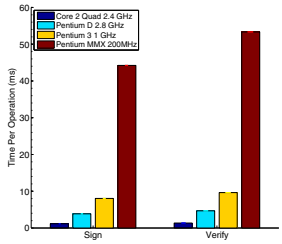
Fig. 3. MAC layer delay

Lankershim data sets, respectively. The highway simulations, I-80, result in significantly worse performance due to the higher density of traffic.

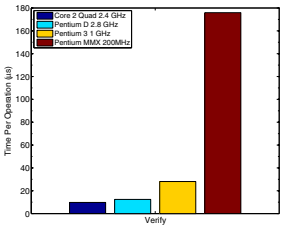
Figure 2 shows that using TESLA always results in a lower fraction of network-layer receptions for any distance as compared to using ECDSA. TESLA performs poorer than ECDSA because the overhead of sending two packets for the non-attached keys simulations increases congestion. For TESLA with keys attached to the following broadcast ("TESLA attached"), the fraction of packets received from the network layer is higher because of the smaller packet size as compared to ECDSA due to signatures being smaller and because there is no separate packet released for keys as with the "TESLA" series. In conclusion, TESLA does not result in reduced channel congestion compared to ECDSA, unless keys are attached to their following broadcast packets.

D. MAC Layer Delay

Figures 3(a) and 3(b) show the CCDF of the MAC layer delay for our I-80 and Lankershim simulations, respectively.



(a) ECDSA



(b) TESLA

Fig. 4. Authentication mechanism verification latency for various hardware

Thus, the graph shows the fraction of packets (y-axis) that have a given latency or longer (x-axis).

These plots show TESLA with separate key release packets resulting in noticeably worse performance in terms of MAC layer delay. In the I-80 simulations, using TESLA with separate key release packets results in 1% of packets being delayed at the MAC layer longer than 16.9 ms, whereas 1% of packets in ECDSA are delayed longer than 5.2 ms. TESLA with attached keys has lower latency than ECDSA because of the smaller packet size. These graphs and the network reception graphs in Figure 2 indicate that additional traffic, e.g., traffic besides heartbeats, will have a significant detrimental effect on the performance of safety applications. Additionally, 1609.4 channel switching [7] will also only increase congestion.

E. Verification Latency

Since vehicles in our simulations need to verify the signatures of a large number of packets, the computational overhead of the signing algorithm used might be non-trivial. From a different point of view, we wanted to investigate the hardware required by vehicles so that they can validate packet signatures in a timely manner. Additionally, we must consider that TESLA requires two packets to be correctly received for a single safety message to be correctly verified: the safety message itself and a key.

In order to investigate the computational power required by vehicles to perform validations while keeping up with packet receptions, we simulated verifying both ECDSA and TESLA signatures on various desktop computer processors. To empirically measure the time taken to sign and verify safety message broadcasts, we wrote a program to calculate signatures over random data that was the size of safety messages. For ECDSA signatures, we used the 224-bit curve

978-1-4244-4148-8/09/$25.00 ©2009

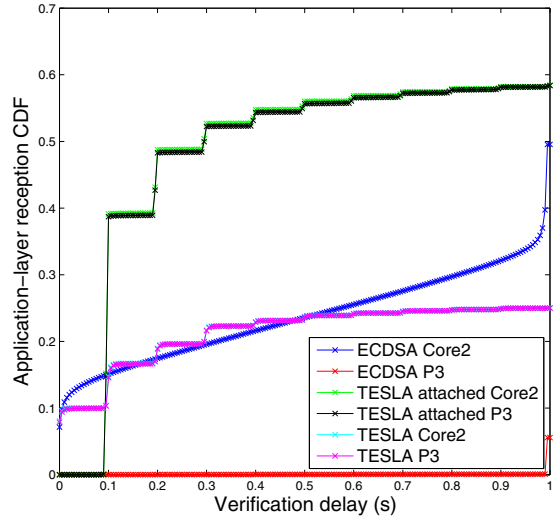
provided in the OpenSSL library [12]. For TESLA signatures, we used the HMAC and SHA-1 functions also provided in the OpenSSL library. For each trial, we verified 1,000 signatures, which were generated over random data the length of a safety message. On each machine we tested, we ran 100 trials, and we calculated the average and 95% confidence intervals for the data. We ran our tests on Pentium MMX 200 MHz, Pentium 3 1 GHz, Pentium D 2.8 GHz, and Core 2 2.4 GHz desktop machines.

Figures 4(a) and 4(b) show the time required to perform signing and verification operations using ECDSA and TESLA signatures, respectively. (The error bars, which represent the 95% confidence intervals, are too small to be noticeable.) Figure 4(a) shows time in *milliseconds*, and Figure 4(b) shows time in *microseconds*. These figures show that verifying an ECDSA signature takes approximately 2-3 orders of magnitude more time than verifying a TESLA signature, independent of the hardware used.

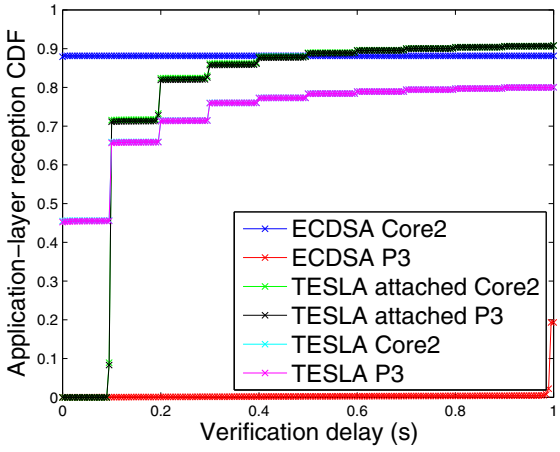
We simulated vehicles with the processing power of both a Core 2 2.4 GHz processor and a Pentium 3 1 GHz processor. The Core 2 processor represents a state-of-the-art processor, and the Pentium 3 represents more accurately the state-of-the-art automotive embedded processor. We allowed vehicles to queue packets for verification purposes but limited the total packet delay to 1.0 second. If the delay was longer than this time, we consider the packet to have been dropped because of a lack of freshness, as many other position broadcasts will have been made by the transmitting node during that time.

Figures 5(a) and 5(b) show the cumulative distribution of total packet verification latency. For a given number on the x-axis (time), the number on the y-axis is the number of packets received with verification latency less than or equal to the number on the x-axis divided by the number of position packets that could have been received at the network layer. The total packet latency is measured from the the time of reception at the PHY layer to the time when signature verification completes and is shown sampled every 5 milliseconds. Measuring this delay allows us to fairly compare ECDSA and TESLA authentication mechanisms because the TESLA delay due to waiting for keys in later safety messages is included, and the ECDSA verification delay due to limited processing power is included.

Figure 5(a) shows that approximately 15% of all receivable safety messages are verified within 0.1 seconds when using a Core 2 processor for verifying ECDSA signatures. When using a Pentium 3 processor, fewer than 0.01% of safety messages using ECDSA signatures can be verified in 0.1 seconds or less. Thus, if ECDSA signatures are to be used for authenticating VANET position broadcasts, VANET computational hardware will need to be more powerful than a Pentium 3 1 GHz processor. Vehicles using TESLA with attached keys receive approximately 10% of their packets at the application layer with a latency of 0.1 seconds or less and 39% with a latency of 0.11 seconds or less, independent of which processor is used for verification. Considering the



(a) I-80



(b) Lankershim

Fig. 5. Cumulative distribution of the total position packet latency including verification time

performance difference between ECDSA and TESLA, as shown in Figure 2, and the latency difference, as shown in Figure 5(a), using TESLA and ECDSA results in a similar percentage of packets with latency 0.1 seconds or less, if we use state-of-the-art hardware when using ECDSA and attach signing keys to their following safety messages for TESLA.

Figure 5(b) shows again that using hardware with computational power equivalent to a Pentium 3 1 GHz while using ECDSA as the authentication mechanism for safety messages results in extremely high latency and a large amount of packet drops due to limited computational power. However, due to fewer vehicles being in the simulation area, as is the case for the Lankershim data used in Figure 5(b), using Core 2 2.4 GHz equivalent hardware incurs almost no latency from the time it takes to verify position broadcast signatures.

978-1-4244-4148-8/09/$25.00 ©2009
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2009 proceedings.

The verification latency data from using TESLA with attached keys shows a large jump at 0.1 seconds. This jump corresponds to the key in the next broadcast. Around the jumps there are slight rises in the number of packets with latency around those areas. This behavior is due to MAC-layer back-off. The additional jumps correspond to the keys released for following packets.

In further investigation, we found that the fraction of packets received and able to be verified is slightly higher for ECDSA at longer distances (beyond about 230 m in the I-80 simulations) due to ECDSA signatures not requiring a second packet containing a key. We omit the figures showing this due to space constraints. The performance at longer distances is particularly interesting because receiving packets at longer distances is required for detecting head-on collisions a sufficient amount of time before the collision would occur.

V. Conclusions

VANETs will be an important addition to automobiles that have the potential to greatly increase vehicular safety. It is important to make sure this technology is correctly designed before it is implemented in vehicles so that its potential can be achieved.

We have shown in Section III that our simulator produces results consistent with that of NS-2, the widely accepted standard for VANET simulations. This is a significant addition to the community because of the great improvement in simulation speed, which enables the VANET community to simulate larger and more realistic environments.

Making use of the proposal to use TESLA as a light-weight broadcast authentication mechanism [3], we presented, to our knowledge, the first published simulations of network performance when using TESLA for broadcast authentication in a VANET comparing both separate and attached key releases. These results showed that using TESLA with attached keys results in a higher fraction of successful packet receptions from the network layer because of smaller packet sizes and a higher fraction of successful application-layer packet receptions as compared to using ECDSA signatures, per the IEEE 1609.2 draft standard [2]. TESLA with either key release mechanism benefits significantly from being able to verify packets with keys following the intended key. Using TESLA with separate key releases results in reduced performance due to either or both of two mechanisms: the additional packets increasing network congestion and/or successfully verifying a safety message requires correctly receiving two packets, the safety message and the packet carrying the key. TESLA attached also suffers from this latter failure mode. As a result of these loss mechanisms, ECDSA performs better than TESLA at greater distances, which is an important area of performance for detecting head-on collisions.

Finally, we investigated the latency of packets from reception at the PHY layer to when they are verified at the application layer. We profiled a variety of computer hardware to measure individual packet verification times under the three authentication mechanisms and simulated verification delays using this data. Our results show that to enable the use of ECDSA signatures for safety message authentication, processing power equivalent to current state-of-the-art hardware is required, but TESLA should perform well on current embedded processors.

References

[1] Q. Chen, F. Schmidt-Eisenlohr, D. Jiang, M. Torrent-Moreno, L. Delgrossi, and H. Hartenstein, "Overhaul of ieee 802.11 modeling and simulation in ns-2," in *MSWiM '07: Proceedings of the 10th ACM Symposium on Modeling, analysis, and simulation of wireless and mobile systems*, (New York, NY, USA), pp. 159–168, ACM, 2007.

[2] IEEE, *IEEE 1609.2-Standard for Wireless Access in Vehicular Environments (WAVE) - Security Services for Applications and Management Messages, available from ITS Standards Program*.

[3] Y.-C. Hu and K. P. Laberteaux, "Strong VANET security on a budget," *Proceedings of the 4th Annual Conference on Embedded Security in Cars (escar 2006)*, November 2006.

[4] V. S. Communications, "Vehicle safety communications project-final report," tech. rep., April 2006.

[5] M. Fiore and J. Härri, "The networking shape of vehicular mobility," in *MobiHoc '08: Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*, (New York, NY, USA), pp. 261–272, ACM, 2008.

[6] U.S. Department of Transportation - Federal Highway Administration, "NGSIM Project." http://www.ngsim.fhwa.dot.gov, October 2008.

[7] IEEE, *IEEE 1609.4-Standard for Wireless Access in Vehicular Environments (WAVE) - Multi-channel Operation*.

[8] J. Yin, T. ElBatt, G. Yeung, B. Ryu, S. Habermas, H. Krishnan, and T. Talty, "Performance evaluation of safety applications over dsrc vehicular ad hoc networks," in *VANET '04: Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks*, (New York, NY, USA), pp. 1–9, ACM, 2004.

[9] C. L. Robinson, L. Caminiti, D. Caveney, and K. Laberteaux, "Efficient coordination and transmission of data for cooperative vehicular safety applications," in *VANET '06: Proceedings of the 3rd international workshop on Vehicular ad hoc networks*, (New York, NY, USA), pp. 10–19, ACM, 2006.

[10] A. Studer, F. Bai, B. Bellur, and A. Perrig, "Flexible, extensible, and efficient VANET authentication," *Proceedings of the 6th Annual Conference on Embedded Security in Cars (escar 2008)*, November 2008.

[11] A. Perrig, R. Canetti, J. D. Tygar, and D. Song, "The tesla broadcast authentication protocol," *RSA CryptoBytes*, vol. 5, p. 2002, 2002.

[12] "OpenSSL." http://www.openssl.org, September 2008.

978-1-4244-4148-8/09/$25.00 ©2009