Efficient Certificate Revocation List Organization and Distribution

Jason J. Haas, Yih-Chun Hu, and Kenneth P. Laberteaux

***Abstract*—In this paper, we propose a lightweight mechanism for revoking security certificates that is appropriate for the limited bandwidth and hardware cost constraints of a VANET. A *Certificate Authority* (CA) issues certificates to trusted nodes, i.e., vehicles. If the CA looses trust in a vehicle (e.g., due to evidence of malfunction or malicious behavior), the CA must promptly revoke the certificates of the distrusted vehicle. To distribute revocation information quickly even during incremental deployment, we propose that CAs use Certificate Revocation Lists (CRLs). The CRL should be composed in a secure manner, and it should be exchanged in a way such that the CRL is both quickly and widely distributed. We previously proposed a mechanism for the quick distribution of CRL updates that also covers a wide area by using vehicle-to-vehicle (V2V) communication [1]. In this paper, we additionally investigate the performance of V2V communication in partial deployment scenarios, that is, where only a certain percentage of vehicles are equipped with VANET radios. We provide simulation results that show our V2V exchange mechanism is quicker than distributing CRLs or CRL updates through road-side units (RSUs) alone. However, this revocation process, which involves both the CA and vehicles, must conform to the aforementioned bandwidth and hardware restrictions. In this paper, we present mechanisms that achieve the goals of reduced CRL size, a computationally efficient mechanism for determining if a certificate is on the CRL, and a lightweight mechanism for exchanging CRL updates. Additionally, we expand on our previous work [2] to provide privacy to revoked vehicles prior to their revocation.**

***Index Terms*—Vehicular networks, VANET, security, revocation, certificate revocation list, CRL.**

I. Introduction

In a VANET, vehicles will rely on the integrity of received data for deciding when to present alerts to drivers, and further in the future, this data may be used as the basis of control decisions for autonomous vehicles. If this information is corrupted, vehicles may present unnecessary or erroneous warnings to their drivers, and the results of control decisions based on this information could be even more disastrous. Information can be corrupted by two different mechanisms: malice and malfunction. Similarly, vehicles have two defense mechanisms: an internal filter and external reputation information. The former defense mechanism can consist of filters based on physical laws (e.g., maximum braking deceleration, maximum speed, and physical space constraints) [3]. The latter defense mechanism can consist of reports from other vehicles or entities on the validity or trustworthiness of data originating from certain vehicles. In this paper, we focus on the latter defense mechanism.

Information received from corrupted vehicles should be disregarded or not trusted by legitimate vehicles. Otherwise, a malicious vehicle could, for example, obtain a less congested route for itself by overstating the number of vehicles on its desired roadway. Additionally, a corrupted vehicle could trigger erroneous driver warnings to be displayed in other vehicles by falsifying its position information, thereby causing drivers receiving the erroneous warnings to suddenly brake, and triggering accidents that might otherwise not have occurred. IEEE 1609.2, the draft standard for security services in vehicular networks, stipulates that vehicles will be authenticated using certificates issued by a *Certificate Authority* (CA) in a *Public Key Infrastructure* (PKI) setup [4]. Illegitimate vehicles should have these certificates revoked, and the identity of the revoked certificates (although ideally not the identity of the associated driver) should be published and distributed to legitimate vehicles. *Whatever mechanism that is used for distributing revocation information should distribute the information securely, quickly, and broadly in order to limit the amount of damage illegitimate vehicles can do.*

The CA can utilize Road-Side Units (RSUs) to distribute this revocation information. However, the density of RSU placement (i.e., how many RSUs are set up in a certain area) has not yet been determined. In fact, it is likely that RSUs will be sparsely placed, and thus, vehicles may spend significant time outside radio contact of an RSU [5]. Even if RSUs are eventually deployed with sufficient density, *a VANET must be able to operate during stages of incremental deployment*, that is, before a sufficient density of RSUs come online. Consequently, we present our work in this paper in light of the following. *How should information about untrustworthy vehicles be organized and disseminated so that the window of opportunity for these untrustworthy vehicles is minimized, especially in light of sparse infrastructure (e.g., during incremental deployment)?* We consider that the information about untrustworthy vehicles is disseminated in the form of the revocation of untrustworthy vehicles' credentials authorizing them to use the VANET, that is, their certificates.

We propose that revocation information be distributed in the form of a Certificate Revocation List (CRL) in an epidemic manner, that is, through vehicle-to-vehicle (V2V) communication [1]. This epidemic distribution mechanism supplements

Manuscript received 5 January 2010; revised 7 May and 12 July 2010.

J. J. Haas and Y.-C. Hu are with the University of Illinois at Urbana-Champaign (e-mail: jjhaas@gmail.edu, yihchun@illinois.edu).

K. P. Laberteaux is with the Toyota Research Institute (e-mail: klaberte@acm.org).

Digital Object Identifier 10.1109/JSAC.2011.110309.

RSU-based distribution,[1] essentially turning vehicles into mobile distributors, making use of vehicles' mobility. The use of V2V distribution of CRLs addresses incremental deployment issues. We will present results evaluating the performance of our epidemic distribution mechanism through simulation and show that our mechanism provides significant advantages over a RSU-only CRL distribution mechanism in terms of speed and breadth of network coverage, even in partial deployment scenarios.

In order to reduce the potential network and computational overhead imposed by any CRL distribution mechanism, we propose two optimizations for organizing and storing CRL information. Our first optimization generates certificate identifiers for a single vehicle for intervals of time using a secret key s_i and a block cipher (e.g., AES) keyed by s_i or a secure Pseudo-Random Number Generator (PRNG) where s_i is used as the seed. For each interval of time, there is a single s_i, which is drawn from a hash chain to preserve vehicles' privacy prior to their revocation. Either of these mechanisms allows the CA to revoke a vehicle's certificates with a single addition to the CRL (i.e., s_i and i), thus minimizing the size of the CRL and CRL updates. We present a mechanism for a single CA to revoke a vehicle in an efficient manner.

Second, to allow a vehicle to quickly check the validity of a certificate, we propose that certificate identifiers be stored in a Bloom filter, which is a probabilistic data structure (i.e., searching has a non-zero, but ideally small, false positive rate) and has a constant ($O(1)$) cost in terms of computation for searching and storage. This approach was proposed by Raya et al. [6], who reviewed Bloom filter false positive rates, and suggested that Bloom filters can be used for storing CRLs. We address the important question of what computational resources are required for a Bloom filter implementation. Our results show that using a Bloom filter for searching a CRL imposes a small amount of computational overhead. We compare Bloom filters with other deterministic data structures. Finally, we discuss the effects of a Bloom filter's false positive rate and how to alleviate those effects.

We will discuss our proposals as follows. We review related work in Section II. In Section III, we propose our privacy-preserving certificate organization optimization. We propose using Bloom filters as a vehicle's local data structure for storing a CRL and investigate issues related to probabilistic data structures in Section IV. We present our simulation methodology for and evaluation of epidemic CRL passing in Section V. Finally, we conclude in Section VI.

II. Related Work

Studer et al. [7] proposed TACKs for certificate organization and vehicle revocation in a VANET, which we consider to be the most relevant and closely related scheme to the work we propose in this paper. The authors propose using group signatures to provide vehicles privacy from authorities. Specifically, TACKs loads vehicles with a number of certificates for a period of time, after which they will need to be reloaded. During reloading, the reload request is authenticated using a group signature. The Regional Authority (RA) receives this request but cannot identify the exact vehicle from which the request originated because the RA is not the group manager, that is, the RA cannot trace a vehicle's identity based on a group signature. When a reload request is made, the RA forwards the request to the CA, which is the group manager, and which can identify the vehicle. The CA then confirms or denies the request (i.e., whether the requesting vehicle is revoked or not), and returns this information to the RA. If the requesting vehicle is trusted, then the RA issues another batch of short-term certificates, i.e., certificates that have a short lifespan. Thus, revocation is accomplished by not issuing a new batch of certificates to revoked vehicles. In the authors' proposal, RAs are responsible for small areas of the total network so that vehicles need to acquire new short-term certificates when entering an area administered by a different RA. Because revocation is performed in an online manner (i.e., by denying vehicles new short-term certificates) and because of the geographic division of the network among RAs, TACKs requires extensive infrastructure deployment and an online CA. Consequently, TACKs provides a solution for certificate organization and distribution after VANET deployment is well established and infrastructure is broadly deployed. Our solution to certificate organization and distribution focuses on being viable during initial deployment (i.e., when there is little infrastructure).

Raya and Hubaux [8] propose using public key ECDSA signatures and show that computation in a vehicular environment is feasible. The authors concur that RSUs may be sparse, especially during incremental deployment.

Online certificate status protocols (OCSPs) serve as an alternative to distributing CRLs. For an OSCP to be practical for VANET, each vehicle would need a constantly-available connection to the CA. Until DSRC can provide continuous connectivity between every vehicle and the CA, use of an OCSP would require a second wireless technology, e.g., cellular data, to be integrated into the DSRC system.

Papadimitratos et al. [9], propose a mechanism for distributing CRLs from RSUs, where they break the CRL into pieces that are encoded using erasure or fountain codes so that cars that obtain a certain subset of the pieces of the CRL can fully reconstruct the CRL. The authors investigate the bandwidth overhead required by their scheme. However, they do not support their results by vehicle simulation or other empirical data, and their results are based on an assumption of RSU placement density. Specifically, the authors present numerical evaluations of RSUs placed every 1, 2, and 3 km.

III. Certificate Organization

For two vehicles to communicate securely, they must possess a copy of each other's credentials in the form of a certificate. Vehicles can exchange certificates upon first encounter. Additionally, certificates can be pre-installed or updated during production and periodic service appointments, which may potentially reduce the need for certificate exchanges.

[1]The CA may utilize other communication methods to distribute CRLs, such as cellular or WiFi. However, the combining of multiple communication technologies within a vehicle's safety computer is not yet a certainty. Further, if a subset of VANET nodes can receive a CRL via an alternative communication technology, the epidemic distribution method can be used to propagate the CRL to other vehicles that lack this alternative channel.

In a VANET, there is a tension between privacy and authentication. Received safety messages must be trusted, and this requires trusting the sender. However, senders wish to retain as much privacy as possible, which implies a receiver must trust, but not conclusively identify, the sender. We explored this tension in mathematical detail elsewhere [10; 11].

Clearly, if a vehicle V has only one certificate, even if it is identified by a pseudonym certificate ID, P, then tracking the vehicle via passive recording of DSRC messages is possible. An attacker, A, who attacks privacy needs only to link P to V one time (e.g., by listening to messages sent by V near V's home location). Then, by employing listening stations near a location of interest, A can determine if and when V visits that location.

A. Certificate Groups

To make it more difficult for A to track V, V could be issued a group of certificates, each with a different ID. V would then use different certificates at different times and locations. Of course, if A is willing to follow V to collect all (or even many) of V's certificate IDs, multiple certificates will not improve V's privacy. However, V has little hope of retaining location privacy from such a determined attacker, that is, if A tracks V, A no longer needs DSRC certificates to track V's movements.[2]

The certificate IDs issued to V should be very difficult for most entities to associate. From the perspective of V's privacy, it is best if V's certificate IDs appear to be completely random.

To make use of the group of certificates it is issued, V must change the certificate it uses. We define a *certificate transition* as the event when vehicle V switches from one certificate to another. It would be relatively easy for those entities within communication range of V at the time of a certificate transition to link these two certificates to the same vehicle. This linking is made even easier if the messages include information about V's current and future location (as some safety applications require [12]). Observing a large number of V's certificate transitions would allow the creation of a certificate profile for V, even if the true identity of V is not known.

It is useful to consider how many certificates V might be issued. This number should be large enough to make it difficult (for an attacker) to observe a substantial number of V's certificates. This concern is removed if certificates are used only once for a period of time and then never reused. However, the number of certificates issued to a vehicle is limited by storage constraints. If certificates are used only once, then there must be reliable opportunities for the vehicle to replenish its list of certificates, or vehicles must be preloaded with a sufficient number of certificates for the expected vehicle lifetime. On the question of the size of the certificate group, we do not offer definitive recommendations but instead describe the logic that influenced our designs.

B. Size of Certificate Revocation Lists

A recent study found that a typical American spent 15 hours/week in a car [13]. As this included time spent as a driver or as a passenger, suppose that an average US vehicle operates no more than 15 hours/week. If each certificate is used for a single 10-minute period and never reused, then 4660 certificates are consumed each year. Therefore, a vehicle will need approximately 5000 certificates per year (fewer, if they are reused). If an opportunity to replenish a vehicle's certificates occurs at least once every 5 years, a vehicle must be able to store approximately 25,000 certificates. If each certificate is approximately 100 B, this requires a storage size of approximately 2.5 MB, which is on the order of on-chip FLASH memory on currently available automotive embedded processors.[3]

Thus, it is possible that a vehicle will possess 25,000 certificates at one time. It is desirable that these certificate IDs be difficult to relate. However, a CA may need to revoke all certificates held by a given vehicle. If the certificate IDs for the revoked vehicle are truly random, i.e., there is no compact representation for the certificate group, then the CA would need to explicitly identify all 25,000 certificates of the revoked vehicle in the updated CRL. Assuming that certificates can be identified by a 16 B fingerprint (the size of one AES block), to revoke a single vehicle, the CRL would thus grow by 400 kB. Since the CRL must be widely propagated, such growth in the size of the CRL would result in excessive computational overhead, storage requirements, and network bandwidth consumption.

Algorithm 1 Certificate generation algorithm.

Require: $M \equiv$ number of certificates per time interval for this vehicle
Require: $I \equiv$ number of time intervals during a reload period
Begin: certificate loading for vehicle V
$\eta \leftarrow get\_random\_nonce()$
for $i = 1$ to I **do**
  $s_i \leftarrow H^i(\eta)$
  for $r = 1$ to M/I **do**
    $(K_{r,i}{}^+, K_{r,i}{}^-) \leftarrow generate\_public\_private\_key()$
    $sig_{CA,r,i} \leftarrow sign(H(\{E_{s_i}(r), K_{r,i}{}^+\}), K_{CA}{}^-)$
    $Cert_{r,i} \leftarrow \{E_{s_i}(r), K_{r,i}{}^+, sig_{CA,r,i}\}$
    $send(V, \{Cert_{1,i}, \ldots, Cert_{M,i}\})$
  end for
end for
End

C. Privacy-Preserving Certificate Groups

We now present our method for creating a group of certificate IDs for one vehicle V that appear to be unrelated but in fact are related by a single secret value s_i known only to the CA. Only for the certificate loading phase of vehicle operation will the CA be required to be online, which will likely occur in a centralized location (e.g., an automobile factory). We present our proposed method in pseudo-code as Algorithm 1.

To obtain this privacy feature, certificates must include a new field containing a single integer. First, the CA chooses a secret random nonce unique to the requesting vehicle, η. The CA then repeatedly hashes this value I times, generating a hash chain using a cryptographic hash function, where $s_i = H^i(\eta), i \in [1, I]$, and $H^i(.)$ is the hash applied i times. Each s_i is a possible revocation key for the vehicle

[2]A determined attacker can, for example, easily track vehicles using license plates.

[3]The Freescale MPC5554, as advertised in April 2008, has 2 MB of FLASH memory.

being reloaded. Each possible revocation key s_i is used as the key to a *block cipher*, which takes fixed length inputs and permutes them into outputs of the same length, where the particular permutation chosen is based on the key s_i. Alternatively, s_i can be used as the seed for a PRNG rather than the key to a block cipher. Next, the CA creates M public/private key-pairs (above, we argued that M could be 25,000). $(K^+_{r,i}, K^-_{r,i}), r \in [1, M/I]$. Then the CA creates a corresponding group of M/I certificates; for each r the CA creates a certificate, $Cert_{r,i} = \{E_{s_i}(r), K^+_{r,i}, sig_{CA}\}$ where $E_{s_i}(r)$ is defined as the certificate ID and is the value produced by encrypting the integer r using the block cipher with the symmetric key s_i (or for a PRNG the r-th output of the PRNG with seed s_i), and sig_{CA} is the CA's digital signature over $\{E_{s_i}(r), K^+_{r,i}\}$ using the CA's private key. The CA then issues these M key-pairs and associated certificates to vehicle V, either before the vehicle begins its service, or during the replenishing events discussed above.

D. Revocation Method

When the CA wishes to revoke vehicle V, it simply appends the values s_i and i to the CRL that corresponds to the time interval during which the revocation event was reported. We will discuss time intervals in more detail in Section III-F. We describe our proposed CRL delivery method in Section V. When a vehicle receives a new CRL that includes a new secret s_i, it generates each r, which are known to every vehicle, and the following revocation keys $s_j, j \in [i+1, I]$. If a result matches the certificate ID of a certificate in its cache, that certificate is revoked.

To increase the revocation list management efficiency, a Bloom filter [14] can be loaded with revoked certificate identifiers. To do this, a vehicle generates all possible revocation keys $s_j = H^{(j-i)}(s_i), j \in [i+1, I]$. Vehicles can recalculate these additional revocation keys because they are drawn from a hash chain and the CRL specifies i and I (if I is not implicitly known). Additionally, a vehicle generates all revoked certificate IDs $E_{s_j}(r), j \in [i, I]$ given a key s_i. $E_{s_i}(r)$ can be calculated by vehicles holding the CRL since the CRL contains the revocation key s_i and the range of r, M/I, (M/I may not be included if it is constant and therefore implied). The resulting certificate IDs can then be included in a Bloom filter. We will discuss what should be done if a certificate identifier results in the Bloom filter indicating a match in Section IV, since our discussion there concerns design parameters for the Bloom filter. Further algorithm extensions employing Bloom filters can be used to efficiently prune cached certificates upon receiving a new revocation key s_i. We present an analysis of the computational overhead imposed by using Bloom filters in Section IV-A, in light of the discussion of our potential setup in Section III-B.

E. Certificate Group Privacy Properties

One desired privacy property is that it should be difficult for an attacker to relate the certificates belonging to one vehicle (i.e., one certificate group) without knowledge of s_i. To make this association, the attacker must be able to break the block cipher or PRNG used to generate $E_{s_i}(r)$.

Unless the attacker observes vehicle V transition between two certificates, the attacker cannot relate two of V's certificates under practical conditions (i.e., the attacker cannot break the cipher used or track the vehicle with an attached GPS tracking device, etc.). Such an attacker would already have compromised V's location privacy. The attacker, and every other vehicle, will be able to group V's certificates once the revocation key s_i and a time interval i for V are exposed. This quick association allows for space-efficient representation of CRLs.

We omit a more detailed proof of the privacy properties of our certificate identifier generation mechanism due to space restrictions. Interested readers may find this proof in our previous work [2].

F. Backwards Privacy

A CA may specify that time be divided into intervals (our use of the term "time intervals"). The CA can then specify for how many time intervals a vehicle should receive certificates during reloading. In our description of revocation keys above, each s_i would correspond to a single time interval.

Using our proposal for linking certificates with a revocation key, a block cipher (or a secret seed and a PRNG), and a cryptographic hash function, vehicles that are revoked have *backwards privacy* with regard to the messages that they have already signed prior to the time interval for which the revocation key was released (i.e., i). Specifically, a party interested in compromising the privacy of a revoked vehicle cannot link a revoked vehicle's signed messages prior to the time interval for which the revocation key was released since the privacy attacker cannot regenerate any of the revoked vehicle's certificate IDs that were used during earlier time intervals better than using a brute force search. We anticipate that this backwards privacy will be desirable. For example, if a vehicle changes owners and the vehicle is revoked under the new owner, then both the old owner and the new owner would have no privacy in terms of the use of the common vehicle, even though the old owner had nothing to do with the event triggered the vehicle's revocation.

G. Discussion

Combining the proposed revocation key approach with Bloom filters for fast certificate acceptance, our certificate generation and revocation scheme has the following properties:

1) Certificates grow by only a small amount for $E_{s_i}(r)$ (one block; 16 B for AES).
2) All of a vehicle's keys can be represented in the CRL with a small amount of data: the revocation key, s_i, (of size 32 B for SHA-256), the time interval i, the total number of time intervals for which the vehicle was issued certificates I (which could be omitted if all vehicles have the same I), and a count of the number of certificates granted for each key, M (which could be omitted if all vehicles have the same M).
3) Any additional loss in privacy is due only to imperfections in the block cipher (or PRNG), or the cryptographic hash function, and not due to the protocol design itself.

4) With memory usage linear in the number of revoked vehicles, each certificate can be checked in time linear in the number of revoked vehicles (no-precomputation approach).
5) With memory usage linear in the number of revoked keys, certificates can be checked very quickly using a Bloom filter with few false positives. (Bloom filters use memory linear in the number of elements it is meant to hold for a fixed false positive rate and number of index functions.)
6) The network overhead involved in transferring CRL information regarding a single vehicle is constant (independent of the number of certificates that a vehicle holds).

IV. Revoked Certificate Storage

As we discussed in Section III-D, we propose that vehicles use Bloom filters to store all revoked certificates. The Bloom filter is not populated with values of s_i received via CRLs. The CRL will contain revocation keys, s_i, and time intervals, i, while each vehicle will insert certificate IDs, $E_{s_i}(r)$, into its Bloom filter. Thus, vehicles can efficiently check the revocation status of both certificates previously received and certificates they receive in the future. We omit a discussion of when Bloom filters should be regenerated and a related discussion on certificate expiration times due to space limitations, and we refer readers to our previous work [2]. In this section, we discuss the results of our analysis of the performance of practical Bloom filter implementations. Additionally, we discuss the consequences of using a Bloom filter. We focus on the effects of using a probabilistic data structure, that is, when searching a Bloom filter, false positives occur with some probability. We also compare the use of Bloom filters with deterministic data structures.

A. Bloom Filter Performance Analysis

A Bloom filter is a data structure that has constant computational cost ($O(1)$) for insertion and search operations. However, Bloom filters are probabilistic data structures, thus there is a chance that a search may indicate a data item is in the filter when it actually is not. In the context of this section, a false positive occurs when a non-revoked certificate appears to be a revoked certificate, that is, appears to be in the Bloom filter. We propose an approach to addressing the problem of false positives below. A desired false positive rate can be chosen by setting 3 parameters: the number of elements to be inserted (n), the size of the data structure (m, here specified in bits), and the number of hash or index functions used for insertions and searches (k). Using this notation, the false positive rate can be calculated as [15]

$$P(\text{false positive}) = \left(1 - \left(1 - \frac{1}{m}\right)^{kn}\right)^{k} \quad (1)$$

To illustrate, let us assume that 1 in 100,000 vehicles are revoked in a VANET with 100,000,000 vehicles, then there are 1,000 vehicles whose revocation keys should appear on the CRL. If each vehicle possesses 25,000 certificates, as mentioned above, then the CRL should encode 25,000,000 certificate identifiers, thus, $n = 25{,}000{,}000$. If we choose $m = 2^{28}$ bits (32 MB) and $k = 6$, then we obtain a false positive rate of 0.616%, or approximately 1 in every 160 certificates. We believe this false positive rate is acceptable for reasons described below.

TABLE I
Intel Core 2 2.4 GHz insert and search times per certificate

Data Structure	Insert (μs)	Search (μs)
Bloom filter	2.6	1.4
AVL tree	6.8	3.4
Red-black tree	5.6	3.6

To evaluate the computational overhead of using Bloom filters, we implemented a Bloom filter, using $k = 6$ hash functions based on MD-5 for index functions and a vector of $m = 2^{28}$ bits for the Bloom filter storage. The input to our hash functions were randomly generated 128-bit certificate IDs. The hash functions were implemented as follows. First, a random byte repeated 16 times was concatenated with the input certificate ID and with the random byte repeated 16 times again. Next, this step was repeated but using a different random byte. (In a deployed VANET implementation, these two padding bytes would be fixed and be specified in a standard.) Each of these two 48 B strings were hashed separately using MD-5. This concatenation mechanism is derived from previous work by Fan et al. [15]. A single MD-5 output was used as 3 different hash functions, taking the 3 indices from 3 different 4 B blocks of the output. Essentially, MD-5 was called 2 times with 2 different random bytes padding the random certificate IDs. Then three 4 B blocks from each output were used to generate 28-bit indices for the Bloom filter. By reusing the MD-5 outputs, we were able to double the filter's performance. We used the OpenSSL implementation of MD-5 for our tests.[4]

We tested the performance of both inserting certificate IDs into and searching for certificate IDs in the Bloom filter. The certificate IDs consisted of 16 B from a cryptographically strong PRNG, also obtained from OpenSSL. We used a cryptographically strong PRNG to emulate the output of our certificate ID generating scheme, as described in Section III. For each insertion or search test, we inserted or searched for 25,000 certificate IDs, which is the number of certificate IDs a vehicle would possess according to our discussion in Section III-A. We recorded the time taken for each batch of tests (25,000 insertions or searches) to microsecond precision, and we performed each batch of tests 10,000 times on each test machine. For the search benchmarking, half of the identifiers sought were in the filter and half of them were not in the filter.

We performed our tests on three different machines, each of which has different hardware. The first machine had an Intel Core 2 (2.4 GHz), the second machine had an Intel Pentium 3 (1 GHz), and the third machine had an Intel Pentium MMX (200 MHz). We omit the figure showing the performance of the other processors due to space restrictions. Table I shows a summary of the performance for the Core 2 processor. These performance tests indicate that all of the certificates from a

[4]See http://www.openssl.org.

revoked vehicle could be added to a vehicle's Bloom filter in 10's to 100's of milliseconds. Also, searching for a certificate in the CRL is 1000's of times faster than the cost of verifying the CA's signature of a certificate. Both of these operations, Bloom filter search and CA signature verification, should be performed by vehicles when they receive a new certificate. The time to search is slightly faster than the insertion time because a search indicates that a certificate ID is not found as soon as a single hash function output is found to not be in the Bloom filter, thus not all hash functions will need to be calculated.

B. Data-Structures

a) Deterministic vs. Probabilistic: Bloom filters are probabilistic data structures, thus, there is a chance of a false positive when searching them. Consequently, it may be preferable to use a deterministic data structure, such as a linked list or balanced binary tree, to store revoked certificate IDs so that no false positives occur. In the previous section, we concluded that there could be up to 25,000,000 certificates revoked under our assumptions. With 16 B certificate IDs, each vehicle would require 381 MB to store all of the revoked certificate IDs. In our discussion above, we found that a Bloom filter with a false positive rate of 0.616% requires only 32 MB of storage. Using a balanced binary tree to store the revoked certificate IDs, the complexity of searching for a single certificate ID is $O(\log n)$.

To compare the performance of deterministic data structures versus Bloom filters, we implemented inserting and searching for certificates in AVL trees and red-black trees. For each type of tree, we populated the tree with 25,000,000 certificates, then searched for and inserted (in that order) 25,000 certificates each. We only tested the performance of each tree on the Core 2 machine because it was the only machine with enough memory to run the tests without accessing a disk.[5] The search times for the trees we used were more than 2 times slower than the search time required with the Bloom filter we tested. However, on this machine, inserting all of the 25,000 certificates of a revoked vehicle could be done in less than 100 ms. Thus, using a deterministic data structure may not have significant additional processing time overhead, but may be unrealistic for immediate deployment because of the memory requirements.

b) False Positive Rate: Having a false positive rate is acceptable for a number of reasons. A certificate from a non-revoked vehicle could trigger a false positive when using a Bloom filter, that is, the certificate could appear to be revoked though that certificate may not be revoked. A vehicle can avoid triggering a false positive by eliminating any of its own certificates that will trigger a false positive. Before using or transmitting a specific certificate, a vehicle can check to see if the certificate will generate a false positive, given the established CRL known to the network. If the certificate will create a false positive, the vehicle can just discard it before it is ever sent, thus preventing a receiver from generating a false positive, assuming every vehicle uses standardized hash functions for the Bloom filter. The reduction in the number of available certificates from this discarding can be overcome by simply loading a fraction more certificates initially. However, under this approach, it is more important for a vehicle to have an updated CRL, so that the vehicle can know whether a certificate will produce a false positive when it changes certificates.

[5]Our AVL and red-black tree tests used GNU libavl on Linux (see http://www.stanford.edu/~blp/avl/index.html) and required approximately 2 GB and 1.5 GB of RAM, respectively.

TABLE II
BLOOM FILTER PRE-LOADING PARAMETERS

c	Number of certificates desired per vehicle
k, n, m	Bloom filter parameters
l	Number of certificates loaded per vehicle
N	Number of vehicles in the network
f	Fraction of revocable vehicles
p	Fraction additional certificates to load
P_{fp}	Probability of a false positive

We now calculate the fraction of additional certificates that must be pre-loaded on vehicles so that vehicles have a desired number (in expectation) of certificates that do not trigger false positives. The terms we use in our analysis here are summarized in Table II. Suppose, the desired number of certificates per vehicle is $c = 25{,}000$, as we used above. Using the same assumptions as we used in the previous section, we set the number of vehicles in the network to be $N = 100{,}000{,}000$. We increase the fraction of revoked vehicles from our assumptions above to $f = 10^{-4}$ in order for the Bloom filter performance to be more robust to having more revocable vehicles than estimated. We denote the additional fraction of certificates that need to be pre-loaded on vehicles as p.

Assuming certificate identifiers are uniformly and independently distributed among vehicles, the expected number of certificates triggering false positives per vehicle is

$$E[P_{fp}] = P_{fp}(1+p)c = pc \tag{2}$$

We set this expected value to be equal to the number of additional certificates to be loaded on each vehicle (pc).

From Fan et al. [15], we use the approximation for k that minimizes the number of certificates that trigger false positives. We denote this optimized number of hash functions as k^*

$$k^* = \arg\min P_{fp}(k) \approx \frac{m}{n} ln2 \tag{3}$$

We calculate n, the number of entries in the Bloom filter as, $n = Nfl = Nf(1+p)c$. Substituting these equations in for k^* and n in to (1) and rearranging, we get

$$\frac{pc}{(1+p)c} - \left(1 - (1 - \frac{1}{m})^{k^* N f (1+p) c}\right)^{k^*} = 0 \tag{4}$$

Choosing $m = 2^{31}$, we solved Equation (4) numerically to get $p = 1.767\%$, which results in $k^* = 6$. Thus, an additional 442 certificates are required to be pre-loaded on each vehicle to replace certificates that are expected to generate false positives. 442 certificates equates to less than 3 days of usable certificates on a schedule of recharging vehicles' certificates every 5 years, using certificates at the rate of 1 every 10 minutes, as we discussed above. If our assumption of $f = 10^{-5}$ is

TABLE III
SIMULATION PARAMETERS

Number of RSUs (R)	1,4,16,64,256,275,300,325
Association time (t_a)	0.1 s, 1.0 s, 2.0 s, 5.0 s, 10.0 s
V2V communication	Disabled ("No V2V"), enabled ("V2V")
Deployment percentage	1%, 2%, 5%, 25%, 100%

accurate and the parameters (k^* and , m) for $f = 10^{-4}$ are used, the expected number of false positives is negligible.

V. EPIDEMIC CERTIFICATE REVOCATION LIST DISTRIBUTION

As we discussed above, for a CA to revoke a vehicle's certificates, the CA appends the vehicle's revocation key to the CRL. The CA then distributes the CRL so that vehicles can identify and distrust newly revoked vehicles. The distribution should spread quickly to every vehicle in the system.

Previous work assumed that CRLs will be distributed by broadcasting updates from RSUs [16]. Distributing a CRL quickly to all vehicles in the system would require a very large number of RSUs to be deployed and maintained by the CA, incurring substantial deployment costs. Another significant challenge to RSU-only distribution is many vehicles may rarely encounter a RSU providing CRL updates, e.g., in rural areas.

Our proposal improves the distribution speed and breadth of the CRL by using vehicles to distribute the CRL in an epidemic fashion. A CRL update could be broadcast by a small number of RSUs in high vehicle density locations. The RSU then *infects* each passing vehicle with the CRL update. Each infected vehicle, in turn, infects every vehicle it encounters. This infection propagates quickly in an epidemic fashion (as verified by our simulations below).

Vehicles may be equipped with additional communications devices (i.e., cell phone data connections) and will, thus, be able to receive CRL updates directly. However, it may be the case that not every vehicle will be equipped with such secondary channel communications options. Thus, epidemic passing can provide CRL updates to vehicles lacking such secondary communication channel equipment.

A. Methodology

In order to evaluate the performance of V2V epidemic CRL passing compared to the performance of CRLs being distributed by RSUs only, we obtained a realistic trace of car movements for simulation. The trace we used was based on the area surrounding the city of Zurich, Switzerland [17], contains nearly 260,000 vehicles, and covered approximately 354 km x 263 km. In order to handle the massive size of this trace, we built a highly-parallel, custom simulator to track encounters between vehicles. This simulator uses a simple infection criteria, described below. It does not attempt to emulate packet-level communication.

In the simulator, vehicles were given velocities and destinations based on the events in the trace file. The positions of the vehicles were sampled every 0.1 s. Every 0.1 s, a list of neighbors was generated for each vehicle. For the simulations where we disabled V2V communication, the neighbor list consisted of only nearby RSUs. For the simulations where we enabled V2V communication, the neighbor list included nearby RSUs and nearby vehicles. We consider the CRL update to be exchanged if two neighbors (one possessing the CRL update) are within 100 m of each other for at least the simulated association time (t_a), which we varied.

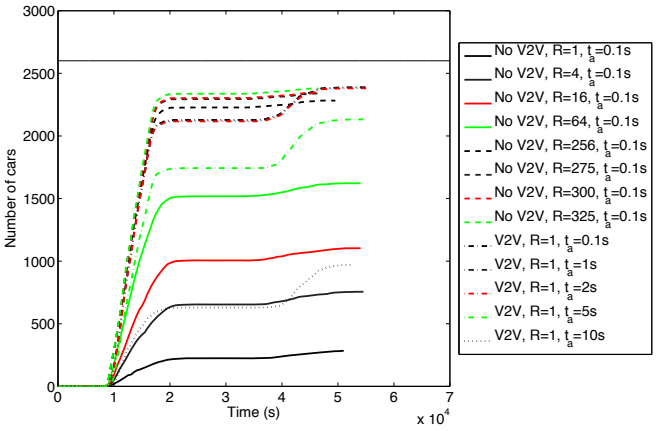


Fig. 1. Number of vehicles possessing a CRL released at 9000 s with 1% deployment

To simulate incremental VANET deployment, we chose various levels (percentage deployed) of vehicles having VANET radios. The average vehicle age in the United States is less than 10 years [18]. Thus, on average, deployment levels should reach 2% in a little under 2.5 months after VANETs begin to be deployed (assuming all new vehicles will be equipped with VANET radios).

In order to determine the locations at which we should place RSUs for maximum effectiveness, we ran simulations without any CRL passing and recorded which grid squares contained the highest number of vehicles at each time step of the simulation. We only recorded the grid squares with 50 or more vehicles in them at any time. We sorted the grid squares based on this count in descending order. In our simulations, we included RSUs based on their order in this sorted list. For example, in a specific scenario, if we used 4 RSUs in our simulation, we chose the 4 RSUs in the grid squares where we recorded the 4 highest numbers of vehicles. Each RSU was placed at the center of its respective grid square. (The size of the grid squares, 100 m x 100 m, with a communication range, 100 m, implies that an RSU is a neighbor of every vehicle in its square.)

In order to validate our simulator, we used a program to display node locations and whether they have the CRL or not at any specified time. Using this visualizer for our simulations, we discovered that there are what appear to be morning and evening rush hours, where vehicles generally move into the city and move out of the city, respectively. Table III summarizes the parameters that we used in our simulations.

B. Results

Figures 1-5 show the total number of vehicles in the simulation that received the CRL versus time, with the CRL update being released at 9000 s. Each of these figures shows

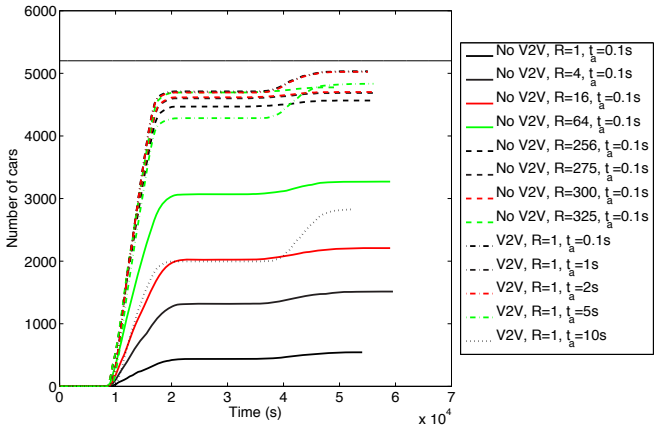


Fig. 2. Number of vehicles possessing a CRL released at 9000 s with 2% deployment

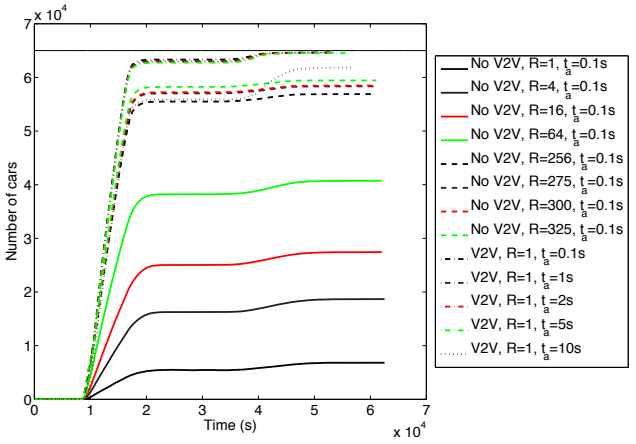


Fig. 4. Number of vehicles possessing a CRL released at 9000 s with 25% deployment

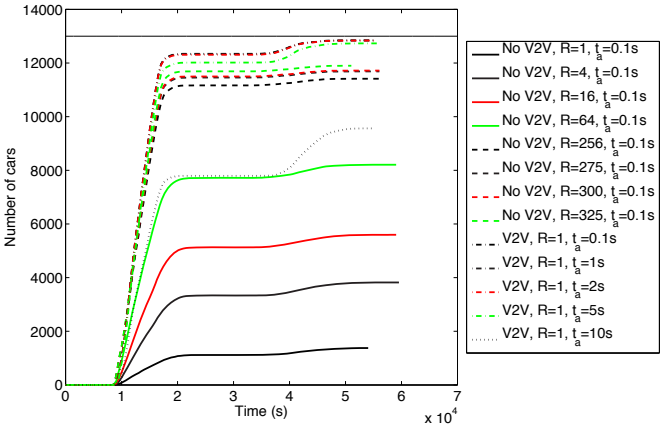


Fig. 3. Number of vehicles possessing a CRL released at 9000 s with 5% deployment

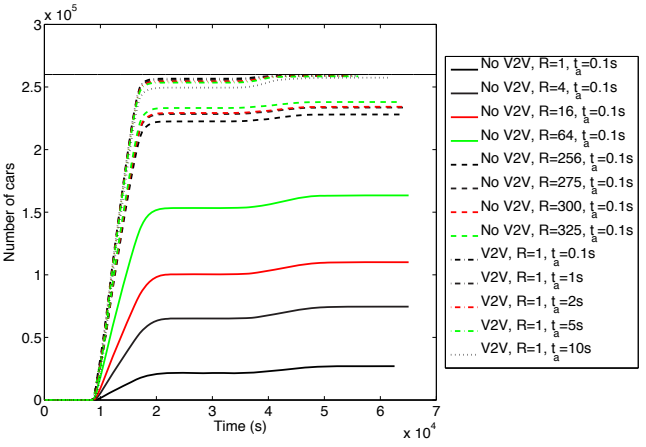


Fig. 5. Number of vehicles possessing a CRL released at 9000 s with 100% deployment

a different level of partial deployment. The case of total deployment (i.e., all vehicles have VANET radios) is shown in Figure 5. Table IV shows the number of vehicles with the CRL update at the end of the simulation as a percentage of the total number of vehicles for each scenario we simulated. To stress the V2V case even further, we used association times of 0.1 s, 1.0 s, 2.0 s, 5.0 s and 10.0 s (all No V2V cases were run with an association time of 0.1 s). Flat spots in each data set occur when there are no new vehicles receiving the CRL update during that time interval. The flat spots between 22,000 s and 32,000 s correspond to a period of time when very few vehicles move in the trace, i.e., between the rush hours. After this plateau, the CRL update distribution progresses again around 35,000s as more vehicles begin to move.

Figure 5 shows that the V2V CRL update passing method with 100% deployment outperforms the method where CRL updates are distributed solely by RSUs in both total coverage (the number of vehicles with the CRL update at the end of the simulation) and speed of coverage (the rate at which vehicles obtain the CRL update) for any of the association times that we simulated (i.e., 0.1 s $\leq t_a \leq$ 10 s). This remains true even when the No V2V case uses 325 RSUs while the V2V case uses only one RSU. In other words, for full deployment scenarios, the case of 1 RSU using V2V passing results in superior CRL update coverage for any length of association time compared to any number of RSUs simulated with V2V CRL update passing disabled.

Figure 1 shows the case of 1% deployment, which results in slightly slower CRL update dissemination for V2V passing and t_a=0.1 s compared to 325 RSUs, but approximately the same number of total vehicles having the CRL update by the end of the simulation. Increasing the deployment to 2%, as shown in Figure 2, results in the V2V dissemination matching the speed of the 325 RSU case but also exceeding the total number of vehicles reached by the 325 RSU case. Figure 3 (5% deployment) shows that V2V dissemination is better than the 325 RSU case with $t_a \leq 5$ s. For higher levels of deployment, the performance of the V2V mechanism continues to outperform the RSU-only mechanism for all of the association times we simulated.

Table IV shows that increasing the association time from 0.1 s to 2.0 s made little impact on the efficacy in passing CRL updates in the V2V cases. V2V passing outperforms RSU-only distribution with the simulated number of RSUs

TABLE IV
PERCENTAGE OF VEHICLES WITH THE UPDATED CRL AT THE END OF THE 76,000 S SIMULATION AND FOR VARIOUS LEVELS OF DEPLOYMENT.

Setup	1% deployed	2% deployed	5% deployed	25% deployed	100% deployed
V2V, R=1, t_a=0.1 s	91.8462%	96.7692%	98.8076%	99.4307%	99.6119%
V2V, R=1, t_a=1.0 s	91.8077%	96.6538%	98.7691%	99.4169%	99.6069%
V2V, R=1, t_a=2.0 s	91.5385%	96.6154%	98.7384%	99.4061%	99.5219%
V2V, R=1, t_a=5.0 s	82.0000%	92.9615%	97.9460%	99.3338%	99.4457%
V2V, R=1, t_a=10.0 s	37.3077%	54.3846%	73.5672%	95.1427%	99.0176%
No V2V, R=325, t_a=0.1 s	91.5000%	91.8462%	91.5378%	91.4501%	91.7035%
No V2V, R=300, t_a=0.1 s	90.2692%	90.4231%	90.1377%	89.9623%	90.0742%
No V2V, R=275, t_a=0.1 s	89.9615%	90.1154%	89.8915%	89.7531%	89.8434%
No V2V, R=256, t_a=0.1 s	87.7692%	87.8077%	87.8068%	87.5252%	87.7220%
No V2V, R=64, t_a=0.1 s	62.3846%	62.8846%	63.1510%	62.6648%	62.9338%
No V2V, R=16, t_a=0.1 s	42.4231%	42.4615%	43.0572%	42.2109%	42.4086%
No V2V, R=4, t_a=0.1 s	29.0769%	29.1154%	29.3946%	28.7391%	28.7428%
No V2V, R=1, t_a=0.1 s	10.9231%	10.4615%	10.5777%	10.4977%	10.4036%

for all deployment levels with $t_a \leq 2.0$ s. As the deployment percentage increases, V2V distribution benefits and can still outperform 325 RSUs alone for longer required association times. It is likely that the association time will be shorter during the initial months of VANET deployment (e.g., 1–5% deployment) because the channel will be less congested.

VI. CONCLUSIONS

We have made two contributions in this paper. First, we proposed a certificate organization method where certificates for a single vehicle are related by a single, secret revocation key. Without this key, certificates are difficult to group, thereby preserving the privacy of a vehicle. However, a revoked vehicle's certificates can be easily identified once the revocation key is distributed via a CRL. To revoke a new vehicle, the CRL need only increase in size by one revocation key and the interval for which that key was used, regardless of the number of certificates provided to the revoked vehicle. We presented specific privacy properties of this scheme, including preserving revoked vehicles' privacy for messages released prior to their revocations. We showed the performance of our certificate identifier storage mechanism, a Bloom filter, which has relatively small memory and computational power requirements.

We also proposed that vehicles be employed to spread CRL updates in an epidemic fashion. Our simulation results showed the performance obtained from using epidemic V2V passing of CRLs obtains better performance for a single deployed RSU than the performance of 325 RSUs without epidemic V2V passing of CRLs. In other words, for deploying RSUs, the cost savings of using a system with epidemic V2V CRL passing is greater than a factor of 325 when compared to an RSU only system, and *can be used effectively to cover an entire metro-area*.

Together, these contributions demonstrate that a lightweight privacy-preserving method for VANET security is possible, even in the case of sparse roadside infrastructure. Furthermore, our proposed schemes are designed to work even during incremental deployment.

REFERENCES

[1] K. P. Laberteaux, J. J. Haas, and Y.-C. Hu, "Security certificate revocation list distribution for VANET," in *VANET '08:Proc. fifth ACM international workshop on VehiculAr Inter-NETworking*, (New York, NY, USA), pp. 88–89, ACM, 2008.

[2] J. J. Haas, Y.-C. Hu, and K. P. Laberteaux, "Design and Analysis of a Lightweight Certificate Revocation Mechanism for VANET," in *VANET '09:Proc. Sixth ACM International Workshop on VehiculAr Inter-NETworking*, ACM, September 2009.

[3] P. Golle, D. Greene, and J. Staddon, "Detecting and correcting malicious data in VANETs," in *VANET '04:Proc. 1st ACM international workshop on Vehicular ad hoc networks*, (New York, NY, USA), pp. 29–37, ACM, 2004.

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

[5] R. Resendes, "The New 'Grand Challenge' — Deploying Vehicle Communications." Keynote Address — The Fifth ACM International Workshop on VehicluAr InterNETworking (VANET 2008), September 2008.

[6] M. Raya, P. Papadimitratos, I. Aad, D. Jungels, and J.-P. Hubaux, "Eviction of misbehaving and faulty nodes in vehicular networks," *IEEE J. Sel. Areas Commun.*, vol. 25, pp. 1557–1568, Oct. 2007.

[7] A. Studer, E. Shi, F. Bai, and A. Perrig, "TACKing together efficient authentication, revocation, and privacy in VANETs," in *SECON'09: Proc. 6th Annual IEEE communications society conference on Sensor, Mesh and Ad Hoc Communications and Networks*, (Piscataway, NJ, USA), pp. 484–492, IEEE Press, 2009.

[8] M. Raya and J. Hubaux, "The security of vehicular ad hoc networks," in *Workshop on Security of ad hoc and Sensor Networks*, 2005.

[9] P. P. Papadimitratos, G. Mezzour, and J.-P. Hubaux, "Certificate revocation list distribution in vehicular communication systems," in *VANET '08: Proc. fifth ACM international workshop on VehiculAr Inter-NETworking*, (New York, NY, USA), pp. 86–87, ACM, 2008.

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

[11] J. J. Haas, Y.-C. Hu, and K. P. Laberteaux, "The Impact of Key Assignment on VANET Privacy," *Security and Communication Networks 2009*, September 2009.

[12] Vehicle Safety Communications, "Vehicle safety communications project-final report," tech. rep., April 2006.

[13] P. Bouvard, L. Rosin, J. Snyder, and J. Noel, "The arbitron national in-car study," tech. rep., December 2003.

[14] B. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Communications of the ACM*, vol. 13, no. 7, pp. 422–426, 1970.

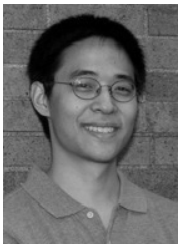
[15] L. Fan, P. Cao, J. Almeida, and A. Z. Broder, "Summary cache: a scalable wide-area web cache sharing protocol," *IEEE/ACM Trans. Netw.*, vol. 8, no. 3, pp. 281–293, 2000.

[16] "Vehicle Infrastructure Integration Concept of Operations, http://www.vehicle-infrastructure.org/program-information, 2006."

[17] V. Naumov, R. Baumann, and T. Gross, "An evaluation of inter-vehicle ad hoc networks based on realistic vehicular traces," in *MobiHoc '06: Proc. 7th ACM international symposium on Mobile ad hoc networking and computing*, (New York, NY, USA), pp. 108–119, ACM, 2006.

[18] U.S. Department of Transportation, Research and Innovative Technology Administration, Bureau of Transportation Statistics, "National transportation statistics," tech. rep., 2009.

Jason J. Haas (M'03) received his bachelor of science degree from the University of Wisconsin-Madison in 2003 in electrical and computer engineering and in physics. He received his master of science degree and PhD from the University of Illinois at Urbana-Champaign in electrical and computer engineering in 2007 and in 2010, respectively. His research interests include network security, vehicular networking, physical security, and cyber-physical systems.



Yih-Chun Hu (M'05) received the B.S. degree in computer science and pure mathematics from the University of Washington, Seattle, in 1997 and the Ph.D. degree in computer science from Carnegie Mellon University, Pittsburg, PA, in 2003.

He is an Assistant Professor in the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Urbana. In his thesis work at Carnegie Mellon Univerity, he focused on security and performance in wireless ad hoc networks. After receiving the Ph.D. degree, he worked as a Postdoctoral Researcher at the University of California, Berkeley, doing research in the area of network security. His research interests include systems and network security.

Ken Laberteaux is a Senior Principal Research Engineer for the Toyota Technical Center in Ann Arbor, MI. Ken's research focus various aspects of sustainable transportation, with a focus on vehicle-grid issues and US Urbanization patterns. Ken has previously worked on information-rich vehicular safety systems, focusing on architecture, security, and protocol design for vehicle-to-vehicle and vehicle-to-roadside wireless communication. He was a founder and two-year (2004, 2005) General Co-Chair of the highly-selective, international Vehicular Ad-hoc Networks (VANET) workshop. Ken completed his M.S. (1996) and Ph.D. (2000) degrees in Electrical Engineering from the University of Notre Dame, focusing on adaptive control for communications. In 1992, he received his B.S.E. (summa cum laude) in Electrical Engineering from the University of Michigan, Ann Arbor.