Secure message propagation protocols for IoVs communication components

1.Motivation:

- Road accidents cause 1.35 million deaths annually.
- Driver irresponsibility identified as the main cause (78%).
- Potential to avoid 60% of accidents with driver warnings.

Paper Roadmap:

1. Introduction:
  - Focus on secure message propagation in IoVs.

2. Related Work:

  - Thorough review of authentication and communication research.
  - Examined VANETs, IoTs, and WSNs.
  - Existing schemes discussed:
    - Privacy-preserving pseudonym generation scheme (Li et al., 2015).
    - Two-factor lightweight scheme relying on system key (Wang et al., 2016).
    - Untraceable and temporal credential-based authentication (Jiang et al., 2016).
    - WSN-based smart vehicular system (Mohit et al., 2017).
    - Lightweight mutual authentication protocol for IoT (Li et al., 2018).
    - Dual authentication and privacy-preserving scheme (Liu et al., 2017).
    - Authentication protocol for IoVs (Chen et al., 2019).
    - Five-layer IoV architecture lacking security parameters (Kaiwartya et al., 2017).

3. IoVs Protocols:
  - Presentation of protocols with main phases:
    - Set-up, registration, authentication, and communication.

4. Security Analysis:
  - Evaluation of the proposed protocols.

5. Protocol Implementation:
  - Practical implementation of the protocols.

6. Performance Evaluation:

- Demonstration of the effectiveness of the protocols.

7. Conclusion and Future Work:
- Final remarks on the research.
- Direction for future work.

- Road Safety Statistics:
- WHO report (2018) states 1.35 million annual deaths globally.
- Main cause: Driver irresponsibility (78%).
- Potential to prevent 60% of accidents with driver warnings.

- Research Focus:
- Secure message propagation in IoVs.

- Paper Roadmap:
1. Section 2: Review of related work.
2. Section 3: Presentation of IoVs protocols (set-up, registration, authentication, communication).
3. Sections 4 and 5: Security analysis and protocol implementation.
4. Section 6: Performance evaluation of protocols.
5. Conclusion and Future Work in Section 7.

Certainly! Here are the key points summarized in bullet form:

3. Proposed Secure Protocols for IoVs Components:

3.1. Network Model:

- Two-layer framework:
IoVs components (low level) and Vehicle Server (VS) (high level).
- Registration requires secure communication via wired or wireless connections using

Transport Layer Security (TLS) protocol.

- Assumptions:
  1. Only registered vehicles can participate.
  2. VS is a trusted entity with high storage and computational capabilities.
  3. OBUs and other entities have storage and processing capabilities.
  4. Registered users do not share passwords with untrusted individuals.

3.2. Steps in Proposed Protocols:

- **Phases involved:** Initial setup, registration, authentication, and communication.
- Table 1 summarizes the notations used.

3.2.1. Initial Setup:
- VS registers all communication components.
- Precomputed key: KVS = h(Sv||IDVS).

3.2.2. IoVs Communication Components Registration:
I. Vehicle Registration:
1. Vehicle (Va) selects IDVa, PWVa, and za.
2. Va computes Ua = h(IDVa||za) and Wa = h(PWVa||za).
3. Ua and Wa sent to VS through a secure channel.
4. VS computes Za = h(Ua||Wa)KVS and sends it back to the vehicle.
5. Vehicle stores parameters in smart card: {SCVa = Za, Ua, Wa}.

II. Road Side Unit (RSU) Registration:
- RSU sends IDRSUb to the VS.

(Note: The remaining steps are not mentioned in the provided text.)

2. For security purposes and registration in IoV (Internet of Vehicles) components, the following steps are taken:

- RSUb Registration:
  - VS generates a nonce (nb) and computes Kb = h(IDRSUb || nb) KVS.
  - VS sends Kb and nb to RSUb, which stores them.

- Portable Device (Mobile) Registration:
  - Mobile device sends IDMc to VS.
  - VS generates a nonce (ic) and computes Gc = h(IDMc || ic) KVS.
  - VS sends Gc and ic to the mobile device (Mc), which stores them.

- Wireless Sensor Device Registration:
  - Sensor device sends IDSd to VS.
  - VS generates a value (pd) and computes Nd = h(IDd || pd) KVS.
  - VS sends Nd and pd to the sensor device through a secure channel.

  - The sensor device stores Nd and pd.

- Infrastructure Registration:
  - Infrastructure sends IDISf to VS.
  - VS generates a value (xf) and computes Qf = h(IDISf || xf) KVS.
  - VS sends Qf and xf to RSUb.
  - RSUb stores Qf and xf.

3.2.3. Authentication:
- Before communication, vehicles must prove their authenticity.
- Every vehicle needs to successfully authenticate itself before sending requests to other entities.

3.2.4. Communication:
- The communication protocols are explained in this section.
- A flowchart (Fig. 7) illustrates the overall idea.
- Inputs for the communication include identity, password, and a random number.

- The communication process involves authentication checks, elimination of non-registered vehicles, data requesting phases, freshness verification, key generation, data reply encryption, forwarding of encrypted reply, key extraction, reply decryption, and integrity checks.

- Vehicle-to-Vehicle (V2V) Communication:

  - When a vehicle (Va) wants information from another vehicle (Ve), it goes through several phases as shown in Fig. 2.

1. Data Requesting Phase:
- Va computes KVS, Aa, and Ba.
- KVS is calculated as KVS = Za h(Ua||Wa) from smart card values.

- Aa is computed using hash operations and **EXOR** operations.
- Va sends Aa, Ba, and T1 to Ve.

2. Identifying Freshness of Message & Request Extraction Phase:
- T1 is calculated based on the request receiving time T2 to verify freshness.
- Ve calculates KVS using stored parameters in the smart card.
- Pa is computed as Aa h(KVS||T1).
- Request Mreqst is extracted as Ba Pa KVS.

3. Key Generation Phase:
- Key Ce is generated as h(Pa||T1||KVS).
- Key De is computed as Ce KVS Pa.

4. Encryption of Reply:
- Ve finds Mrply and **encrypts** it as EMrply = ENCCe(Mrply).
- EMrply, De, and T2 are sent to Va through an **insecure channel.**

5. Identifying Freshness of Message & Key Extraction Phase:
- T2 is compared with T3 − T2 to verify freshness.
- Va computes two different keys using **transmitted parameters** and their own parameters.
- If the computed keys match, further communication proceeds.

6. Reply Extraction Phase:
- Va extracts the reply as Mrply = DECCe(EMrply), where EMrply is the encrypted form of the reply message.

II. Vehicle to Portable (Mobile) Device (V2P) Communication:

1. Data Requesting Phase:
- Va computes KVS, Ja, La, and Ia.
- KVS is calculated using stored values.
- Ja, La, and Ia are calculated based on hash operations and request message.
- Va sends Ia, Ja, and T4 to Mc.

2. Identifying Freshness of Message & Request Extraction Phase:
- Mc verifies the freshness of the message based on T4 and T5.
- KVS is computed using stored parameters.
- ra and La are recomputed based on hash operations.
- Request Mrq is extracted as Ia La KVS.

3. Key Generation Phase:
- Key Nc is generated based on computed values.
- Key Oc is calculated as Nc KVS La ra.

4. Finding Reply and Encryption Phase:
- Mc finds the reply Mrp and encrypts it using key Nc as EMrq = ENCNc(Mrp).
- Mc sends EMrq, Oc, and T5 to Va.

5. Identifying Freshness of Message & Key Extraction Phase:
- T5 is compared with T6 − T5 to verify freshness.
- If the message is not old, Va extracts the key and checks for further communication.

III. Vehicle to RSU (V2R) Communication:

1. Data Requesting Phase:
- At time T7, Va computes KVS, Ea, Fa, and Ga.
- KVS is computed using stored values.
- Ea, Fa, and Ga are calculated based on hash operations and request message.
- Va sends Ea, Fa, Ga, and T7 to RSUb.

2. Identifying Freshness of Message & Request Extraction Phase:
- RSUb verifies the freshness of the message based on T7 and T8.
- KVS is computed using stored parameters.
- qa and Fa are recomputed based on hash operations.
- Request Mreq is extracted as Ga Fa KVS.

3. Key Generation Phase:
- Key Hb is generated based on computed values.
- Key Ib is calculated as Hb KVS Fa qa.

4. Encryption of Reply:
- After receiving Mreq, RSUb finds the reply Mrply.
- The encrypted reply is sent to Va as EMrply = ENC(Hb, Mrply).
- RSUb sends EMrply, Ib, and T8 to Va.

5. Identifying Freshness of Message & Key Extraction Phase:
- Va checks the freshness of the message by comparing T7 and T8 - T7.

- If the message is not old, Va computes the key in two different ways, one from the transmitted parameters and the other from its own parameters.
- Nc = h(qa||T7||KVS||Fa) and Ib = Hb KVS Fa qa.
- Va checks the values of Nc and Ib. If they are the same, further communication proceeds.

6. Reply Extraction Phase:
- Va extracts the reply Mrp as Mrp = DEC(Hb, EMrply), where EMrply is the encrypted form of the reply message.
- This ensures secure communication between Va and RSUb.

To continue:

IV. Vehicle to Infrastructure (V2I) Communication:

1. Data Requesting Phase:

- At time T10, Va computes KVS, βa, Xa, and Ya.
- KVS is computed using stored values.
- βa, Xa, and Ya are calculated based on hash operations and request message.
- Va sends βa, Xa, Ya, and T10 to ISf.

2. Identifying Freshness of Message & Request Extraction Phase:
- ISf checks the freshness of the message based on T10 and T11.
- KVS is computed using stored parameters.
- ta and Xa are recomputed based on hash operations.
- Request Mrqst is extracted as Ya ta KVS.

3. Key Generation Phase:
- Key Af is generated based on computed values.
- Key Uf is calculated as Af KVS Xa ta.

4. Encryption of Reply:
- After receiving Mrqst, ISf finds the reply Mrpy.
- The encrypted reply is sent to Va as EMrpy = ENCAf(Mrpy).
- ISf sends EMrpy, Uf, and T11 to Va through an insecure channel.

5. Identifying Freshness of Message & Key Extraction Phase:

- Upon receiving the values of EMrpy, Uf, and T11, Va checks the novelty based on T11 and T12 - T11.
- If the message is not old, Va computes the key in two different ways, one from the transmitted parameters and the other from its own parameters.
- Af = h(ta||T10||KVS||Xa) and Af = Uf KVS Xa ta.
- Va checks the values of Af and Af. If they are the same, further communication proceeds.

6. Reply Extraction Phase:
- Va extracts the reply Mrpy as Mrpy = DECAf(EMrpy).
- This ensures secure communication between Va and ISf.

V. Vehicle to Wireless Sensor (V2S) Communication:

1. Data Requesting Phase:
- At T13, Va computes the values of KVS, Pa, Qa, and Ra.
- KVS is computed using stored values.
- Pa, Qa, and Ra are calculated based on hash operations and request message.
- Va sends Pa, Qa, Ra, and T13 to Sd.

2. Identifying Freshness of Message & Request Extraction Phase:
- Sd verifies the freshness of the message based on T13 and T14.
- KVS is computed using stored parameters.
- sa and Qa are recomputed based on hash operations.
- Request Mrequest is extracted as Ra sa KVS.

3. Key Generation Phase:
- Key γd is generated based on computed values.
- Key Dd is calculated as γd KVS Qa sa.

4. Encryption of Reply:
- After getting Mrequest, Sd finds Mreply.
- The encrypted reply EMrply is sent to Va as EMrply = ENCγd(Mreply).
- Sd sends EMrply, Dd, and T14 to Va through an insecure channel.

5. Identifying Freshness of Message & Key Extraction Phase:
- Upon receiving the values of EMrply, Dd, and T14, Va checks the novelty based on T14 and T15 - T14.

- If the message is not old, Va computes the key in two different ways, one from the transmitted parameters and the other from its own parameters.
- γd = h(sa||T13||KVS||Qa) and γd = Dd KVS Qa sa.
- Va checks

the values of γd and γd. If they are the same, further communication proceeds.

6. Reply Extraction Phase:
- Va decrypts Mreply with the key γd as Mreply = DECγd(EMrply).
- This ensures secure communication between Va and Sd.

The proposed protocols are resilient against various attacks in vehicular systems:

4.1. Impersonation: The protocols make it computationally infeasible for an adversary to impersonate a legitimate user due to the use of hash functions and time-stamped requests.

4.2. Modification: Comparison checks ensure the integrity of transmitted parameters, allowing detection of any modifications made to the data.

4.3. Replay Attacks: Freshness checks based on timestamps and time differences prevent the acceptance of replayed messages, effectively countering replay attacks.

4.4. Eavesdropping: Encryption techniques safeguard sensitive information, making it inaccessible to eavesdroppers who may intercept the communication.

4.5. Denial of Service (DoS) Attacks: While the protocols do not directly address DoS attacks, implementing rate limiting, authentication, and access control mechanisms can mitigate the impact of such attacks.

In summary, the proposed protocols provide strong security measures against impersonation, modification, replay attacks, and eavesdropping. Additional measures can be employed to address DoS attacks and further enhance system security.

4.3. Replay Attacks: The proposed protocols include initial checks on the validity of received requests using time-stamp values. If the time-stamp value exceeds a predetermined threshold, the processing cannot proceed, effectively preventing replay attacks.

4.4. Password Guessing: To determine the validity of a guessed password, the protocols require the calculation of specific values (e.g., Ua and Wa) derived from the hash of the identity, password, and nonce. If the computed values do not match the stored values, the primary check fails, and communication is terminated. As a result, the proposed protocols are resistant to password guessing attacks.

4.5. Man-in-the-Middle Attacks: While a man-in-the-middle attack is possible if an attacker can understand the parameters transmitted over a public channel, the proposed protocols mitigate this risk. The protocols compute a key that can only be calculated by the communicating parties, who possess the necessary credentials. This ensures that even if an attacker intercepts and gains knowledge of the values exchanged, they cannot compute the key and successfully carry out a man-in-the-middle attack.

In summary, the proposed protocols provide protection against replay attacks, password guessing attacks, and mitigate the risk of man-in-the-middle attacks in vehicular systems.

5. Protocol Implementation

5.1. Implementation on a Single Desktop Computer:

- Windows 10 operating system
- Intel Core i5-7200U processor with 2.50 GHz clock
- 8.1 GB memory
- Various operations used: hash function (Th()), asymmetric encryption (Tae), asymmetric decryption (Tad), symmetric encryption (Tpe), symmetric decryption (Tpd), and signing operation (Tsg)
- Timings for operations:
  - Th(): 0.0020 ms
  - Tae: 4.4063 ms
  - Tad: 7.7613 ms
  - Tpe/pd: 0.0100 ms
  - Tsg: 24.8350 ms

5.2. Implementation on a Raspberry Pi:

- Raspberry Pi model: BCM 2708 System-On-Chip

- ARMv6-compatible processor rev 7
- 8 GB SD card
- Code implemented in Python 3.6
- Use of SHA 2 module from hashlib module for secure cryptographic hash function
- Timings for operations:
  - Th(): 0.1740 ms
  - Tae: 866.7330 ms
  - Tad: 2686.5330 ms
  - Tpe/pd: 1800.0000 ms
  - Tsg: 709.1490 ms

Additional information:
- The increased computation time on the Raspberry Pi is due to its lower processing power compared to the desktop computer.
- Refer to Fig. 8 for the Raspberry Pi setup and connections.

6. Performance Analysis

6.1. Communication Cost & Storage Overhead:
- Communication cost: The total number of bytes transferred is considered.
- Storage overhead: The total number of bytes stored is considered.

- Parameters and their sizes:

  - SHA-2 hash digest: 256 bits (32 bytes)
  - Identity/random number size: 10 bytes
  - Timestamp size: 8 bytes
  - Asymmetric encryption/decryption size: 128 bytes
  - Symmetric encryption/decryption size: 16 bytes
  - Signature size: 192 bytes

- Definitions:

  - Gh(): Hash operation
  - Gid: Length of identity
  - Gts: Length of timestamp
  - Gaenc/adec: Asymmetric encryption/decryption
  - Gpenc/pdec: Symmetric encryption/decryption
  - Gsg: Signing operation
- Refer to Table 2 for the parameters used in the proposed protocols and existing VANETs, IoTs, and WSNs.

6.2. Computation Time and Battery Consumption:
- Computation time: Calculated based on the total number of required cryptographic operations in the login, authentication, and communication phases.
- Operations like hash, signing, encryption, decryption, etc. are executed around 1000 times.
- Negligible time for certain operations compared to others.
- Battery consumption: Total energy required for the complete communication process.
- Computed as BCP = TET * CMP, where BCP is the battery consumption power, TET is the total required experimental time (computation time), and CMP is the CPU maximum power for wireless communications.
- General value of CMP for wireless systems.

Note: Detailed results and analysis can be found in the respective research paper.

6. Performance Analysis

6.1. Communication Cost & Storage Overhead:
- Communication cost: Consider the total number of bytes transferred.
- Storage overhead: Consider the total number of bytes stored.
- Parameters and their sizes: Refer to Table 2 for specific sizes of parameters.
- Definitions: Gh() - Hash operation, Gid - Length of identity, Gts - Length of timestamp, Gaenc/adec - Asymmetric encryption/decryption, Gpenc/pdec - Symmetric encryption/decryption, Gsg - Signing operation.

6.2. Computation Time and Battery Consumption:

- **Computation time:** Calculated based on the total number of required cryptographic operations in the login, authentication, and communication phases.
- Certain operations have negligible time compared to others.

- **Battery consumption:** Computed as BCP = TET * CMP, where BCP is the battery consumption power, TET is the total required experimental time (computation time), and CMP is the CPU maximum power for wireless communications.

- **General value of CMP for wireless systems:** 10.88 W for a Desktop Computer [32] and 2.5 W for a Raspberry Pi [33].

6.3. Results Analysis:

- Comparative analysis of secure authentication and communication-based VANETs, IoTs, and WSNs research works.
- The proposed protocols show lower communication and storage costs compared to competitive schemes.
- **Wang et al.'s scheme** [23] has faster computation time but has disadvantages mentioned in the related works section.

- Graphical comparison of communication cost and storage overhead values.
- Performance analysis considers the type of parameters, their lengths, and computation time.

- Battery consumption depends on the computation time.
- Final remarks based on the performance analysis.

7. Conclusions and Future Scope:
- First effort towards designing secure protocols for IoVs communication components.
- Design based on **cryptographic operations**.
- In-depth security analysis shows resistance against various adversarial attacks.
- Implementation on a **Desktop Computer and Raspberry Pi**.
- Better performance in communication cost, storage overhead, computation time, and battery consumption compared to other schemes.
- Lightweight nature allows easy incorporation into different entities.
- Future work includes secure protocols for other Smart city entities and designing a **secure global protocol for all IoVs communication components.**

Note: For more details and specific findings, refer to the research paper.