Lightweight Security Scheme for Vehicle Tracking System Using CoAP

Arijit
Ukil,
Innovation Lab
Tata Consultancy
Services, Kolkata,
India
arijit.ukil
@tcs.com

Soma
Bandyopadhyay,
Innovation Lab
Tata Consultancy
Services, Kolkata,
India
soma.bandyopadh
yay@tcs.com

Abhijan
Bhattacharyya,
Innovation Lab
Tata Consultancy
Services, Kolkata,
India
abhijan.bhattachar
yya@tcs.com

Arpan
Pal
Innovation Lab
Tata Consultancy
Services, Kolkata,
India
arpan.pal@tcs.co
m

ABSTRACT

In this paper we present a lightweight security scheme for authentication and key management to establish a secure channel for Intelligent Transportation System (ITS) for an IoT (Internet of Things) application. We choose Constrained Application Protocol (CoAP) as lightweight application layer protocol. Low overhead security is still an open challenge for CoAP. We propose a payload embedded low cost symmetric-key based robust authentication and key management mechanism on CoAP. This minimizes the security overhead by eliminating expensive handshaking and ciphersuite agreement of standard TLS and DTLS. We propose some unique modification in the CoAP header to invoke its secure mode in an optimized manner. Further, we propose a secure channel with adaptive reliability which reduces the overall communication cost. Such a low overhead security scheme for CoAP is hitherto unexplored. The efficacy of our proposed scheme is demonstrated through laboratory experiments in an emulated environment.

Categories and Subject Descriptors

C.2.1 [Computer-Communication Networks]:Network Architecture and Design – wireless communication; H.m [Information Systems]: Miscellaneous

General Terms

Algorithms, Performance, Design, Theory, Experimentation, Security.

Keywords

IoT, Security, Sensors, CoAP, Lightweight.

1. INTRODUCTION

The paradigm of IoT has ushered in a new promise through a realization and possibilities of a plethora of applications such as ITS, E-health and SHM (smart home management) for smarter universe. An IoT system typically consists of resource constrained sensing devices which sense environmental attributes and transfer the sensed data to a backend infrastructure over Internet using a sensor gateway. Sensors and senor gateways are usually constrained devices. Hence it remains a challenge to communicate sensor data in a bandwidth and energy efficient, yet secure manner.

In this paper we consider the use case of ITS which tracks the vehicle by fetching GPS coordinates of the vehicle's position and monitors its speed by obtaining accelerometer data of the vehicle. Here in-vehicle sensor gateway posts vehicle-specific sensed data such as GPS coordinates, and accelerometer data, using CoAP [1]. The retrievable information, such as the current location of the vehicle and possible time to reach a certain location, may be very sensitive for the vehicle owner or driver. Different malicious agents [2] can attack such sensitive information. Thus it is essential to satisfy three requirements for sensor data transactions: 1) authentication, 2) encryption, 3) whole transactions and their processing must be lightweight in terms of resource consumption. In fact, existing secure wireless protocols do not provide solutions to reduce this substantial overhead [3, 4].

In this work, we endeavor to embed a low overhead security mechanism consisting of both authentication with integrated key management and encryption on CoAP [5]. The proposed mechanism is robust against different security-breaching attacks such as chosen plaintext attack, replay attack and man-in-the-middle attack. Our proposed security scheme leverages the request-response layer of CoAP. A novel approach is designed to enable secure mode of CoAP by introducing a unique option in CoAP header. It further adapts handshaking level of its secure channel depending on the state of vehicle (like moving fast, moving slowly, at rest etc.). The main aim of this adaptation is to further reduce the communication cost in terms of bandwidth and power consumption. The efficacy of our claims is shown in terms of latency, total bandwidth consumption and computation cost by analyzing experimental results obtained in an emulated environment.

The paper is organized as follows. In Section 2, we analyze the state-of-the-art. The system architecture is described in Section 3. We present threat modeling and security engineering in Section 4. In Section 5, our proposed low overhead security scheme, its security analysis is described and its implementation for CoAP, for the vehicle tracking application is illustrated. In Section 6, we discuss the experimental results and analysis. Finally we conclude our paper in section 7.

UbiComp'13 Adjunct, September 8–12, 2013, Zurich, Switzerland.
Copyright 2013 ACM 978-1-4503-2543-1/13/09…$15.00.
http://dx.doi.org/10.1145/2523501.2523504

2. RELATED WORK

Web-enablement of constrained sensor and gateways using traditional HTTP based protocol would be unsustainable and non-scalable. CoAP is established as candidate lightweight protocol [6] for Internet connectivity of such energy-constrained sensors and it is evident from table 1 [1]. The authors in [7] have shown improved performance of CoAP against HTTP. However, ensuring low overhead security on CoAP is a challenging task. A two-party secure communication protocol for constrained devices using elliptic curve cryptography is described in [8]. This is a puzzle solver authentication known as Host Identity Protocol Diet EXchange (HIP DEX). Currently, the trend of using security scheme for sensor devices is based on symmetric key [4, 9]. Another approach is based on DTLS (Datagram Transport Layer Security), a datagram counterpart of TLS [10].Such efforts do not suit well constrained sensor devices due to the computational overhead of public key cryptosystem that use PKI-based certification and communication overhead of a lengthy handshaking process. However in order to adapt DTLS to constrained devices, low overhead security schemes such as RawPublickey without having any x.509 certification are introduced using preconfigured public keys [5]. However, DTLS has at least 25 bytes overhead per packet that carries a fragment and fills around 1/3 of the usable frame-size [11]. [4] described a mechanism for low overhead message integrity checking using MAC (Message authentication code) stripping. [19] proposed a mechanism for protocol characteristics adaptation based on sensed indication derived from the vehicle's state, but it does not define the use of secure channels with adaptive reliability.

Table 1. Resource consumption comparison between HTTP and CoAP [1]

Parameters	HTTP	CoAP
Bytes per transmission	1451	154
Power(mw)	1333	151
Lifetime (days)	0.744	84

In this paper, we focus on introducing a low overhead security mechanism using symmetric-key based authentication and confidentiality for CoAP suitable for constrained sensor devices implemented with a practical application. We also propose a scheme for adapting reliability of a secure channel, for further reduction of communication overhead. As far as our knowledge goes no such attempt has been made on low overhead security on CoAP applied on a use case like that mentioned above.

3. SYSTEM ARCHITECTURE

We present a secure vehicular tracking system. Each vehicle is equipped with Digi ConnectPort-X5 M2M gateway embedded with multiple sensors like GPS, gyroscope and accelerometer [16].This gateway uses GPRS connection to post the vehicle tracking information. It updates the GPS co-ordinates along with its current speed inferred from the collected accelerometer data periodically to a back-end server using the proposed secure CoAP. The vehicle tracking application is deployed on backend server. A remote user can track the vehicle graphically using this application.

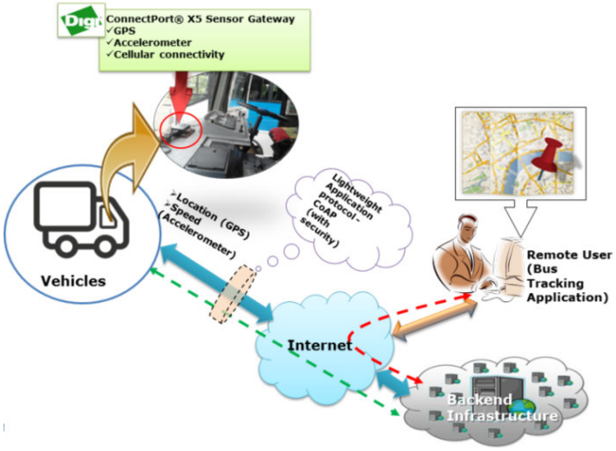

Digi
ConnectPort® X5 Sensor Gateway
✓GPS
✓Accelerometer
✓Cellular connectivity
Vehicles
➢Location (GPS)
➢Speed
(Accelerometer)
Lightweight Application protocol-CoAP (with security)
Remote User (Bus Tracking Application)
Internet
Backend Infrastructure

Figure 1. System architecture of a typical vehicle tracking system.

4. THREAT MODEL AND SECURITY ENGINEERING

Our security engineering is an iterative process to enforce fine-grained adjustments on security primitives (key length, algorithm) with resource and security requirements as shown in figure 2.

We assume eavesdroppers, message interceptors in broadcast wireless channel with substantial capability of replay attack, man-in-the-middle attack, chosen plaintext attack (CPA). Also it is assumed that nodes are hardware tamper-resistant such that security primitives cannot be compromised.

We consider a bottom up approach for meeting limitation of resources. This allows stable and effectively engineered secure system for considered use case. The security threats are disclosure of sensitive information and resource consumption attacks.

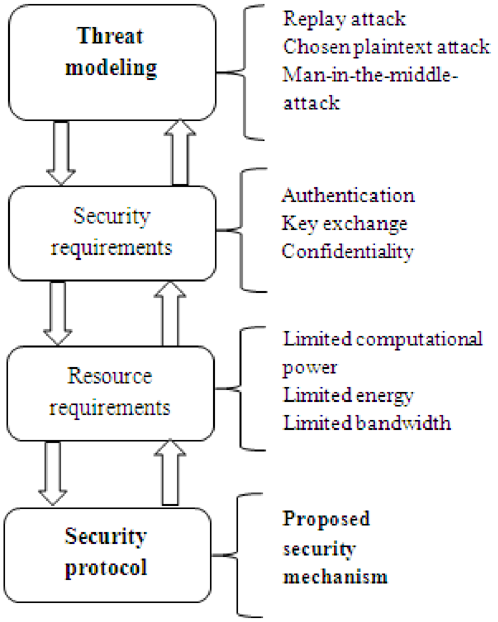


Figure 2. Security threat model and security engineering.

5. PROPOSED METHOD

5.1 Authentication Mechanism

Our proposed security solution is symmetric key based authentication with integrated key management. Exchanged symmetric key is used with AES 128 CBC (Cipher Block Chaining) mode, NIST recommended ciphering scheme [4]. This method is payload embedded thus minimizing the handshaking overhead. It consists of following phases:-1) secret distribution, 2) session initiation, 3) server challenge, 4) sensor response. This scheme is orders of magnitude faster than conventional PKI-based systems because of the absence of any public-key crypto component. This solution eliminates hazards of complicated key management as described in [5]. Our proposed method is a two round trip process as compared to CoAP+ DTLS [11], which has at least four round trips. Also, our proposed method does not have any mutual agreement on cipher suites. We assume that during provisioning phase (at manufacturing or deployment of sensors) CoAP-enabled sensor devices are equipped with security information like keying information as described in [5]. At the time of provisioning of a sensor gateway and server a unique secret is pre shared, which in our case is considered as hardcoded with the device at the time of manufacturing and deployment. In order to secure the authentication scheme against the threats described earlier, we propose nonce based authentication-key management [13]. We have followed negotiation and challenge-response processes in our proposed scheme, which is described below and shown in figure 3. List of notation is shown in Table 2.

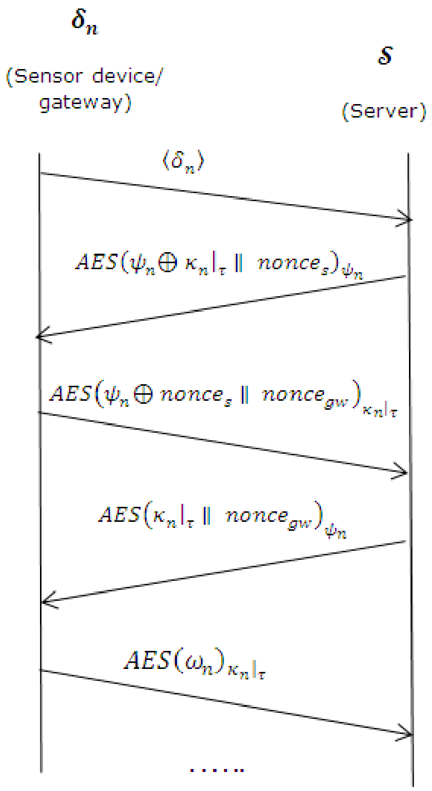

δ_n
(Sensor device/ gateway)
$\mathcal{S}$
(Server)
$\langle\delta_n\rangle$
$AES(\psi_n \oplus \kappa_n|_\tau \parallel nonce_s)_{\psi_n}$
$AES(\psi_n \oplus nonce_s \parallel nonce_{gw})_{\kappa_n|_\tau}$
$AES(\kappa_n|_\tau \parallel nonce_{gw})_{\psi_n}$
$AES(\omega_n)_{\kappa_n|_\tau}$
……

Figure 3. Proposed security protocol.

1. $\mathrm{M}_1|_{\delta_n \rightarrow \mathcal{S}}$: δ_n (sensor gateway) initiates and sends $\langle\delta_n\rangle$ its intention of communication with $\mathcal{S}$ (server)
2. $\mathrm{M'}_1|_{\mathcal{S} \rightarrow \delta_n}$: $\mathcal{S}$ verifies the validity of$\langle\delta_n\rangle$. If it is valid, $\mathcal{S}$ responds as $AES(\psi_n \oplus \kappa_n|_\tau \parallel nonce_s)_{\psi_n}$,$|\mathrm{M'}_1|_{\mathcal{S} \rightarrow \delta_n}| = \{0,1\}^{256}$, where $\kappa_n|_\tau$ is randomly generated key by $\mathcal{S}$ at each re-authentication or re-keying, which is evoked when time-out interval of the authentication session expires. $\psi_n = \{0,1\}^{128}$, the shared unique secret.
3. $\mathrm{M}_2|_{\delta_n \rightarrow \mathcal{S}}$: δ_n extracts $\kappa_n|_\tau$ and $nonce_s$ from$\mathrm{M'}_1|_{\mathcal{S} \rightarrow \delta_n}$as $\psi_n \oplus \kappa_n|_\tau \oplus \psi_n = \kappa_n|_\tau$. It also returns $AES(\psi_n \oplus nonce_s \parallel nonce_{gw})_{\kappa_n|_\tau}$
4. $\mathrm{M'}_2|_{\mathcal{S} \rightarrow \delta_n}$: $\mathcal{S}$ checks $nonce_{\mathcal{S}}$ from$\mathrm{M}_2|_{\delta_n \rightarrow \mathcal{S}}$, if successful sensor is authenticated. $\mathcal{S}$ then returns $AES(\kappa_n|_\tau \parallel nonce_{gw})_{\psi_n}$ to get itself authenticated to gateway.
5. $\mathrm{M}_3|_{\delta_n \rightarrow \mathcal{S}}$: δ_n checks $nonce_{gw}$ from$\mathrm{M'}_2|_{\mathcal{S} \rightarrow \delta_n}$ if$\mathcal{S}$ is successfully authenticated, authentication process completes.

Confidentiality: After successful bi-directional authentication,δ_n sends sensor data ω_n to $\mathcal{S}$ as: $AES(\omega_n)_{\kappa_n|_\tau}$

Table 2. Notation

Notation/ Symbol	Meaning
ψ_n	Shared secret between sensor gateway δ_n and server $\mathcal{S}$
$\kappa_n|_\tau$	Key exchanged between sensor gateway δ_n and server at τ^{th} session
$\langle\delta_n\rangle$	Unique sensor device/ gateway ID of δ_n
$AES(.)_\kappa$	AES operation on plaintext using key κ
$nonce_{i=s,gw}$	$nonce_s$ = server initiated nonce $nonce_{gw}$ = gateway initiated nonce
$\oplus$	XOR
$\parallel$	Concatenation
ω_n	Sensor data of sensor gateway δ_n

5.2 Security Analysis

Following steps summarize the security analysis of our proposed method:

1. Intuitively, proposed authentication-key exchange protocol is semantically secure as per following definition [14]. Let us consider:

$\mathscr{g}$: function about key ($\mathcal{K}_i$) generation (between gateway and server) that the attacker tries to learn.

$\mathscr{h}$: attacker's a priori knowledge on $\mathcal{K}_i$.

$\{\mathbb{K}_p\}_{p\,\in P}$: probability ensemble of key space of $\mathcal{K}_i$, where p is any key among universal key set P.

As defined in [4], in our semantic security model (Φ, Ψ, Ω) [$\Phi :$ key generation function for κ_n, (key) $\Psi, \Omega :$ encryption, decryption on κ_n and $\omega_{\mathrm{n}}(plain\ text)$] with symmetric-key authentication: for each probabilistic polynomial-time algorithm $\mathcal{Z}$ there exists $\mathcal{Z}'$such that for $\{\mathbb{K}_p\}_{p\,\in P}$the space of every polynomial-bound ensembles, every polynomial-bound functions g, h: $\{0,1\}^* \to \{0,1\}^*$, every positive polynomial $p(.)$and all sufficiently large *n*:

$$P_r\left[\mathcal{Z}\left(1^n, \Psi_{\kappa_n\,(1^n)}(\mathbb{K}_p), 1^{|\mathbb{K}_p|}, h\left(1^n, \mathbb{K}_p\right)\right) = g(1^n, \mathbb{K}_p)\right] < P_r\left[\mathcal{Z}'\left(1^n, \quad 1^{|\mathbb{K}_p|}, 1^{|\mathbb{K}_p|}, h\left(1^n, \mathbb{K}_p\right)\right) = g(1^n, \quad \mathbb{K}_p) + \frac{1}{p(n)}\right]$$

2. Proposed nonce-based authentication and key exchange method Ψ is semantically secure under CPA for all attacker $\mathcal{A}$, Advantage of A over Ψ (ADV[$\mathcal{A}, \Psi$]) under CPA iff [15]:

$$ADV_{CPA}[\mathcal{A}, \Psi] = |Pr[EXP\ (0) = 1] - Pr[EXP\ (1) = 1]| \leq \epsilon$$

where, $\epsilon \rightarrow 0$.

Here, $\epsilon \leq 2^{-48}$.Ψ is AES-based in CBC mode, which has cipher gain of2^{32}. It can be proved that Ψ requires key-refreshment of 2^{48} blocks [15].

Proof sketch [5]: For every adversary $\mathcal{A}$ attacking Ψ_{CBC}, there exists a PRP (Pseudo Random Permutation) adversary $\mathcal{B}$ such that:

$$Adv_{CBC}\,[\mathcal{A}, \Psi_{CBC}] \leq 2.Adv_{PRP}\,[\mathcal{B}, \Psi] + \frac{q^2 l^2}{|X|}$$

Security is when $q^2 l^2 \ll |X|$

With AES-128: $|X| = 2^{128}$, $ql < 2^{48}$

Nonces are generally generated using larger length random number generation (RNG) to minimize collision attack. However, in practice, true RNG is difficult to find [12]. Our solution uses a pseudo random number generation (PRNG) appended with a timer (counter). Nonce is non-reproducible due to randomness of $\mathcal{R}_j$ (PRN) along with monotonic incremental nature of $\mathcal{T}_j(timer)$. $\mathcal{R}_j$ is generated in pseudo-random way, and its inclusion with $\mathcal{T}_j$ assures that replay attack is improbable:

$$\left\{Pr\left(R_j\big|_{t=T} = R_j\big|_{t=T'}\right) = 1\right\} < \epsilon',\ \epsilon' \to 0.$$

The predictable non-reproducibility among nonces are governed by $\mathcal{T}_j$ and the non-predictable part is governed by $\mathcal{R}_j$. The birthday bound problem states that when α out of 2^β number of elements are drawn in mutually independent way, ρ_c, the collision probability is upper bounded by [12]:

$$\rho_c|_{max} \geq \frac{\alpha^2 - \alpha}{2.2^\beta}$$

$$2^\beta \geq \frac{\alpha^2 - \alpha}{2.\rho_c|_{max}} \cong 2^\beta \geq \frac{\alpha^2}{2.\rho_c|_{max}}, \text{ when } \alpha \gg 1$$

We have considered, $\alpha = 2^{56}$ and $\rho_c|_{max} = 2^{-56}$

$2^\beta > 2^{111} \to \beta_{min} = 112.$

Thus justifying $\mathcal{R}_j$ = 112 bits and counter $\mathcal{T}_j$ = (128 – 112) = 16 bits.

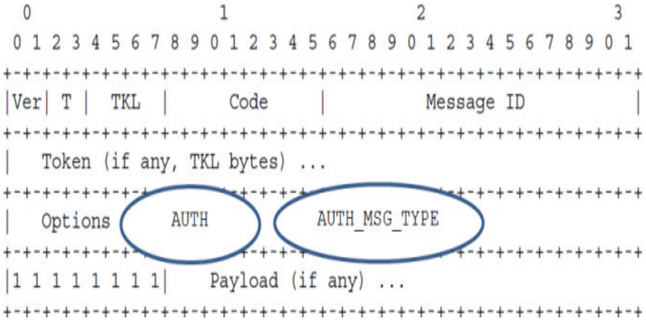
Hence, we prove that our proposed method is secure under the considered threat model as the nonce-respecting scheme is immune to replay attacks, while AES-CBC is resilient to CPA.

5.3 Embedding the Security scheme into CoAP

In this section we present how above the described low overhead security mechanism is embedded into CoAP, thereby proposing a lightweight secure version of CoAP for IoT systems. In our proposed method request-response mechanism of CoAP is applied to establish a secure communication channel. One significant aspect of our proposed scheme is to piggyback the response of authentication with ACK of CoAP's confirmable message. This reduces the number of message transactions between the sensor device/ gateway and server.

5.3.1 Embedding authentication

POST method with confirmable (CON) data transfer mode is applied to achieve mutual authentication between sensor-gateway (client) and backend server. We introduce a new option 'AUTH' in CoAP header to enable the secure (authentication) mode. 'AUTH' uses an unused option indicating a critical option class. Along with 'AUTH' one more option 'AUTH_MSG_TYPE' is introduced to indicate different messages for establishing an authentication session. Figure 4 depicts optional header enabling this secure mode. It is to be noted that the authentication session is maintained by using a constant 'Token' value in header for all the associated messages exchanged during the authentication phase. Proposed authentication session establishment is depicted in figure 5.



```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Ver| T |  TKL  |      Code     |          Message ID           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Token (if any, TKL bytes) ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Options       AUTH          AUTH_MSG_TYPE
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|1 1 1 1 1 1 1 1|    Payload (if any) ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4. Options introduced into the CoAP header to embed the proposed security scheme.

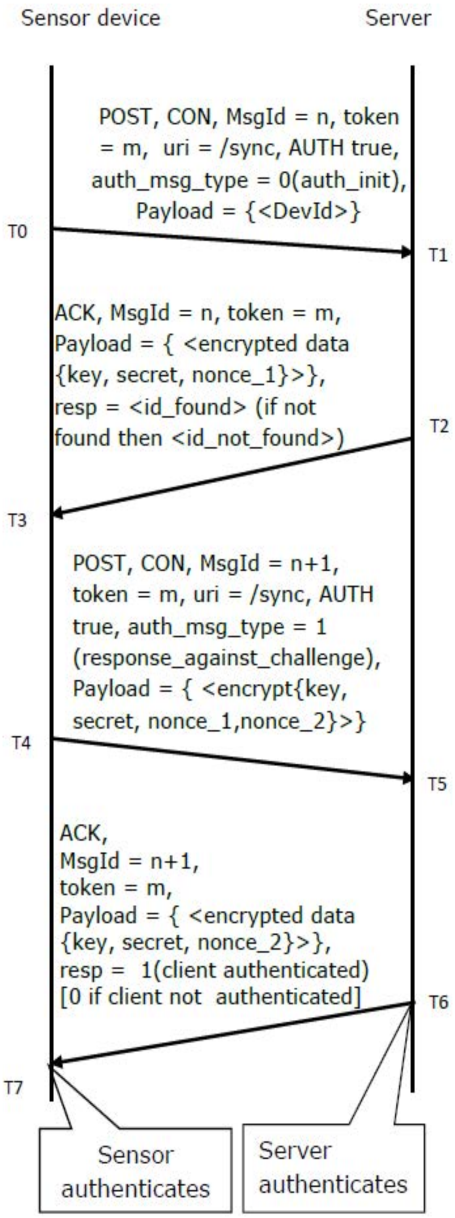

Sensor device
Server
POST, CON, MsgId = n, token = m, uri = /sync, AUTH true, auth_msg_type = 0(auth_init), Payload = {<DevId>}
T0
T1
ACK, MsgId = n, token = m, Payload = { <encrypted data {key, secret, nonce_1}>}, resp = <id_found> (if not found then <id_not_found>)
T2
T3
POST, CON, MsgId = n+1, token = m, uri = /sync, AUTH true, auth_msg_type = 1 (response_against_challenge), Payload = { <encrypt{key, secret, nonce_1,nonce_2}>}
T4
T5
ACK, MsgId = n+1, token = m, Payload = { <encrypted data {key, secret, nonce_2}>}, resp = 1(client authenticated) [0 if client not authenticated]
T6
T7
Sensor authenticates
Server authenticates

Figure 5. Embedding the authentication mechanism on CoAP.

The following steps are performed to embed authentication within CoAP.

1. At initiation sensor-gateway sends a POST message with CON mode having the above mentioned option fields with AUTH_MSG_TYPE value as ‘auth_init’, and ‘device identifier’ in the payload.
2. Server derives device identifier from payload and determines pre-shared secret associated with that device-identifier after receiving options ‘AUTH’, and ‘auth_init’ value for AUTH_MSG_TYPE. It then generates nonce_1 and Key (K). ‘K’ encrypts and decrypts data during confidentiality phase. Server generates an encrypted payload using the shared secret as shown in step 2 of authentication mechanism.
3. Server sends a response back to sensor-gateway with our introduced response code ‘id_found’ with the generated payload as described above. One significant point to be noted here, that server piggybacks this response code and this payload with ACK of received confirmable POST message. In case of an invalid device identifier server sends a response code ‘id_not_found’.
4. Sensor-gateway decrypts response received from server embedded on ACK of its last POST message by using shared secret as stated in step 3 of authentication mechanism, there by obtains nonce_1 and ‘K’. It generates the nonce_2 and then follows step 4 of authentication mechanism to generate encrypted payload by using key ‘K’. It sends this payload using a POST message with option field ‘AUTH’, and AUTH_MSG_TYPE value as ‘response_against_challenge’, and with same token value as in last POST message.
5. Server decrypts payload of above POST with above mentioned optional values in header by using ‘K’ and checks the received nonce_1. Server sends a response with response code ‘client authenticated’ if nonce_1 is identical with its previous value (generated in step 2), otherwise sends ‘client not authenticated’.

Computation time of different phases of authentication shown in table 3 is significantly less in comparison to the retransmission timeout of reliable messages of CoAP considered for our use case. This justifies the piggybacking of authentication payload with response of confirmable message.

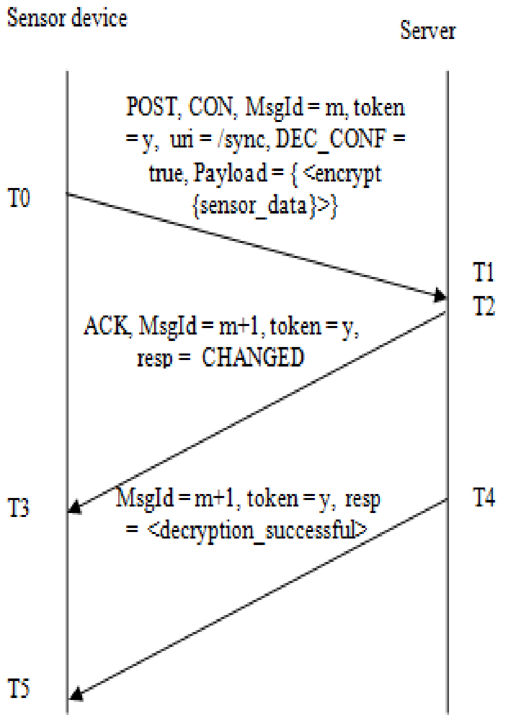

Sensor device
Server
POST, CON, MsgId = m, token = y, uri = /sync, DEC_CONF = true, Payload = { <encrypt {sensor_data}>}
T0
T1
T2
ACK, MsgId = m+1, token = y, resp = CHANGED
T3
MsgId = m+1, token = y, resp = <decryption_successful>
T4
T5

Figure 6. Exchange of data payload through a secure channel after the successful authentication.

Table 3. Computation time (sec) at different phases of authentication at sensor gateway

Atomic operation		$M_2\|_{sensor} \equiv T4 - T3$	$M_3\|_{sensor} \equiv \Delta T7$
Encryption	128 bit	NA	NA
	256 bit	1.372	NA
Decryption	128 bit	NA	.731
	256 bit	1.431	NA
Overall computation time		2.837	.734

5.3.2 *Embedding confidentiality*

After authentication, the encrypted data using 'K' is getting posted by using POST with a newly introduced option type 'DEC_CONF' in header using CON mode. In our case payload (vehicle-tracking information) consists of following fields: <vehicle ID, Route ID, Lat, Long, Time Stamp, Accelerometer Data>. After decryption, server decides to send a response code depending on received value of 'DEC_CONF' as true or false. In case of 'DEC_CONF' false it does not send any response otherwise send a response. This response message consists of a response code indicating the status of success or failure of decryption. According to this status the client resends previous encrypted data. This time also we have piggybacked the response code using the ACK message of the CON mode of POST as the decryption time in the server end is on average 0.67 second obtained from our experimental results significantly less than retransmission timeout. However we propose a separate response path as depicted in figure 6 to send the decryption status when decryption time is significantly large. This is an optional feature for our proposed method and depends on an application's need.

5.4 Adapting the Secure Channel Reliability based upon the Vehicle State

The state of the vehicle signifies whether the vehicle is moving at a low or high speed or in static condition. While sending the vehicle tracking information, the current application also fetches the vehicle state from the local analytics module. The sensor gateway sends an option type 'OMIT_DEC_STAT' int he header of a POST message, along with encrypted payload containing vehicle state information as depicted in figure 7. This time CoAP running in sensor gateway adapts the NON (non-confirmable) mode while sending POST message. The server at other end, after obtaining the encrypted message with option type 'OMIT_DEC_STAT', does not send any decryption status. This time no response message is sent from the server. This further reduces the amount of handshaking. For a vehicle moving at a high speed, the vehicle tracking information is getting refreshed with new values. Therefore, we do not consider whether vehicle information is decrypted successfully at every attempt or not.

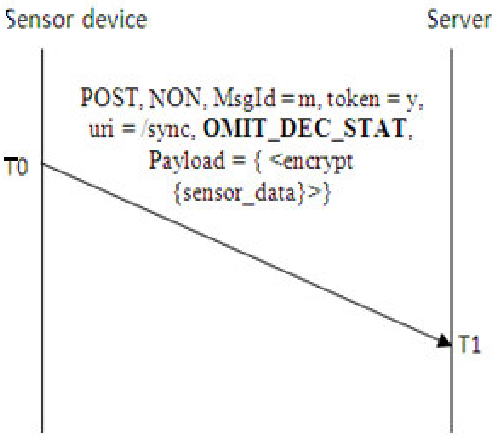


Figure 7. Adaptation of reliability based on vehicle-state. Sensor- gateway adapts non-reliable mode; sends an indication to server to suppress decryption status as well as the response code.

6. EXPERIMENTAL RESULTS AND ANALYSIS

In this section we demonstrate our experimental results. The experimental setup is shown in figure 8. We have used DigiConnectPort X5 [16] as in-vehicle sensor gateway, which is equipped with accelerometer and GPS sensor. The device is enabled with Ethernet, WiFi, ZigBee and Cellular interfaces. Operating system of this device is a small Linux footprint and it is packaged with a customized Python 2.6 library. For laboratory experiment, we use Ethernet connection for device login and interconnection among server, sensor and network emulator. We emulate wireless condition inside laboratory, where network emulator WANEM is used [17].

We consider stringent wireless network condition with 9.6KBps data rate and three types of packet loss: 0%, 10% and 20%. In order to eliminate synchronization issue, we examined closed loop latency. We show performance comparison in figure 9 and it is found that latency overhead for incorporating our method does not exceed 5% when packet loss is 20%. In case of 0% packet loss latency is almost similar. Another performance comparison is bandwidth requirement of our proposed security scheme and normal CoAP. We experimented with two different payloads: one with 30 second data accumulation period (≈ 820 Bytes) and another with 60 second data accumulation period (≈ 950 Bytes). We find that the message size increment in secure CoAP is less than 2% as shown in figure 10.

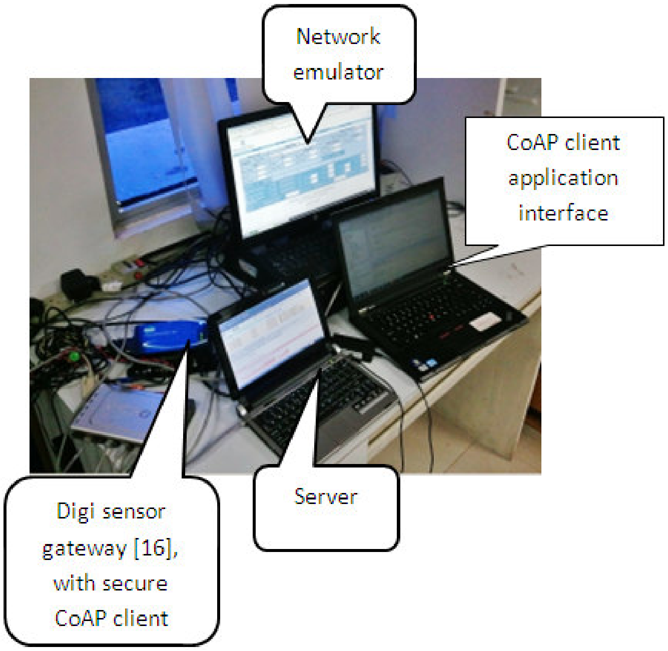

Network emulator
CoAP client application interface
Digi sensor gateway [16], with secure CoAP client
Server

Figure 8. Experimental setup in wireless network emulated environment using DigiConnectPort X5 [16] as sensor gateway with secure CoAP client and WANEM as network emulator, CoAP server running in standard PC.

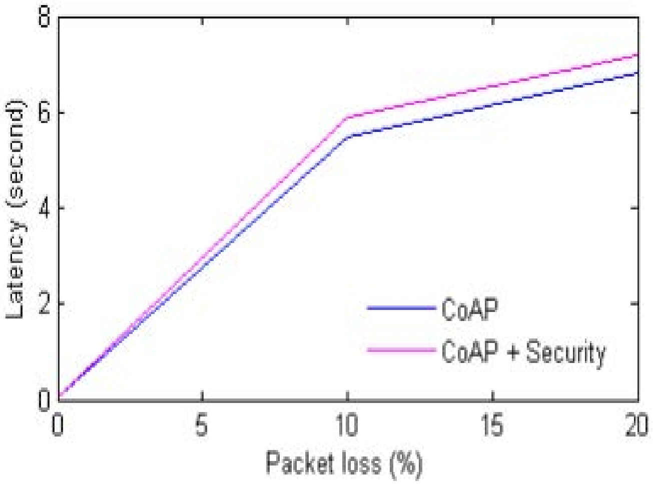


Figure 9. Closed loop latency comparison at different packet loss (%) condition between CoAP and secure CoAP with our proposed mechanism.

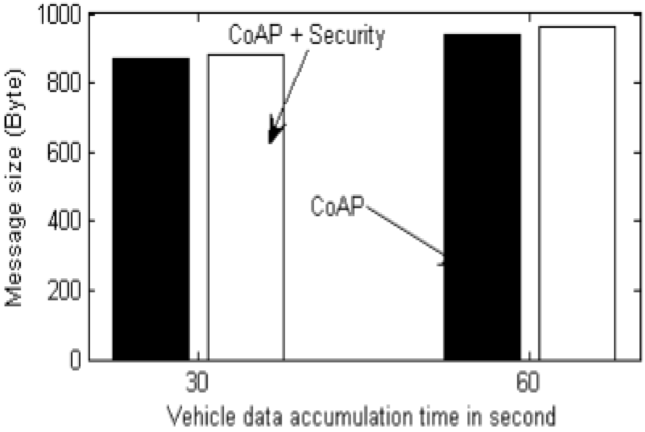


Figure 10. Bandwidth consumption (in bytes) comparison using different data accumulation period (in seconds).

7. CONCLUSION

We have presented a lightweight security mechanism to protect sensor data exchange. We specifically developed a method to enhance CoAP and implemented it for a vehicle tracking application. Our proposed security scheme is resilient to typical security threats in an IoT system [18, 20]. It has a low overhead due to payload embedded symmetric key based authentication with integrated key management. Thus makes it ideal for securing resource constrained sensor devices.

We introduced unique header option in CoAP to establish a secure channel between the sensor gateway and the backend server. The key idea is to design the secure mode of CoAP to be as light as possible. Other novel contribution of our work is to reduce the amount of handshaking for reliability based on the vehicle state information particularly when the vehicle is running at a high speed. The experimental results show that the proposed secure scheme for CoAP improves performance in terms of the security, robustness and resource utilization in typical IoT applications.

8. ACKNOWLEDGMENTS

We thank Sitaram Venkata Chamarty and Praveen Gauravaram of TCS Innovation Lab, Hyderabad, India for their valuable comments and suggestions.

9. REFERENCES

[1] Colitti, W. Steenhaut, K. and Caro, N.D. Integrating Wireless Sensor Networks with Web Applications. In *Proc. IPSN* (2011).

[2] Ukil, A. Context protecting privacy preservation in ubiquitous computing. *In Proc. Computer Information Systems and Industrial Management Applications (CISIM 2010)*, IEEE (2010), 273 – 278.

[3] Li, Y. Li, J. Ren, J. and Wu, J. Providing hop-by-hop authentication and source privacy in wireless sensor networks. *In Proc. INFOCOM 2012,* IEEE (2012), 3071-3075.

[4] Mare, S. Sorber, J. Shin, M. Cornelius, C. Kotz, D. Adapt-lite: privacy-aware, secure, and efficient mhealth sensing. *In Proc. WPES 2011*, ACM Press (2011), 137 - 142.

[5] Shelby, Z. Hartke, K. and Bormann, C. Constrained Application Protocol (CoAP), draft-ietf-core-coap-18, 28 June, 2013.

[6] Bandyopadhyay, S. and Bhattacharyya, A. Lightweight Internet protocols for web enablement of sensors using constrained gateway devices. *In Proc. International Conference on Computing, Networking and Communications (ICNC), 2013*, San Diego, CA, IEEE (2013), 334 – 340.

[7] Bandyopadhyay, S. and Bhattacharyya, A. Energy Efficient Sensor Data Distribution Using Mobile Phone in Cyber-Physical-System*. In Proc. 14th International Conference on Distributed Computing and Networking (ICDCN)*, 2013, Mumbai, India.

[8] Moskowitz,R. HIP Diet EXchange (DEX), IETF draft-moskowitz-hip-rg-dex-01, July 7, 2010.

[9] Eronen, P. and Tschofenig, H. (Editors) Pre-Shared Key Ciphersuites for Transport Layer Security (TLS) (*RFC 4279*).

[10] Modadugu, N. and Rescorla, E. The Design and Implementation of Datagram TLS. *In Proc. NDSS* (2004).

[11] Hartke, K. and Bergmann, O. Datagram Transport Layer Security in Constrained Environments. *draft-hartke-core-codtls-01* (2012). http://www.ietf.org/proceedings/83/slides/slides-83-lwig-2.pdf

[12] Zenner, E. Nonce Generators and the Nonce Reset Problem. *In Proc. 12th International Security Conference (ISC 2009)*, 411 - 426.

[13] Needham, R. M.; Schroeder, M. D. Authentication revisited. *In ACM SIGOPS Operating Systems Review* 21 (1), Jan 1987. doi:10.1145/24592.24593.

[14] Lindell, Y. Foundations of Cryptography. Dept. of Computer Science Bar-Ilan University, Israel (2010).

[15] Boneh, D. Stanford University, 2012, http://crypto.stanford.edu/~dabo/cs255/lectures/PRP-PRF.pdf.

[16] http://www.digi.com/products/wireless-routers-gateways/routinggateways/connectportx5#overview.

[17] Kalitay, H.K. and Nambiarz, M.K. Designing WANem: A Wide Area Network Emulator tool. *In proc. 3rd International Conference on Communication Systems and Networks (COMSNETS)*, 2011.

[18] Ukil, A. Security and Privacy in Wireless Sensor Networks. *In Book Smart Wireless Sensor Networks*, Intechweb Press (2010), 395 – 418.

[19] Bandyopadhyay, S. Bhattacharyya, A. and Pal, A. 2013. Poster Abstract: Adapting Sensed Indication for Vehicular Analytics. *In proc. SenSys*, Nov 11-15 2013, Roma, Italy (to appear)

[20] Ukil, A. Sen, J. and Koilakonda, S. 2011. Embedded Security for Internet of Things. *In Proc. 2nd IEEE National Conference on Emerging Trends and Applications in Computer Science (NCETACS)*, 2011, India.