Research Article

International Journal of
Distributed
Sensor Networks

International Journal of Distributed Sensor Networks
2017, Vol. 13(9)
© The Author(s) 2017
DOI: 10.1177/1550147717731041
journals.sagepub.com/home/ijdsn
SAGE

Improve security of wireless sensor networks through reluctant checksum

Qiong Zhang[1] and Jianyu Xiao[2]

Abstract

The open wireless channel makes the data prone to being eavesdropped. Current wireless security schemes are designed to protect data through cryptography. But an adversary can still recover the secrets by eavesdropping the frames and performing off-line brute-force attacks. Capturing valid frames is fundamental for such attacks. It is generally accepted that each frame at data link layer is designed to include cyclic redundancy check (CRC) checksum sequence for integrity check. However, frame checksum sequence also helps adversary to capture correct frames and drop corrupted ones, which are fundamental to future off-line attacks. In this article, we argue that from the perspective of wireless security, it is unreasonable to include checksum sequence in data link layer frame without any protection. A reluctant checksum scheme named R-CS is proposed. Based on the inherent characteristic of wireless networks that frame error is inevitable, checksum of frame is protected by accumulated checksum algorithm in R-CS. The checksum of the frame cannot be decoded by any nodes except the receiver. Without checksum, adversaries cannot distinguish error frames from correct ones. R-CS requires little computation and communication resources, which is particularly suitable for resource-limited wireless sensor network. Our experimental results clearly demonstrate that R-CS is feasible for wireless sensor network.

Keywords

Wireless sensor networks, security, cumulative checksum, open channel problem

Date received: 30 June 2017; accepted: 18 August 2017

Academic Editor: Shancang Li

Introduction

In the past few years, we have witnessed a rapid penetration of wireless networks into the home and enterprise. With the increased usage of wireless networks, wireless network security receives more concern. Wireless networks are vulnerable to attacks which are more difficult to launch in the wired domain due to the broadcast nature of the wireless channel. The attackers have easy access to the wireless channel.[1] They can eavesdrop and capture network frames for future attacks, causing the open channel problem. Open channel problem is difficult to solve. Take WLAN (wireless local area network) as an example, WPA2 (Wi-Fi Protected Access 2) is widely used because it provides strong encryption. But the attackers can still gain unauthorized access to WLANs by frame capturing and off-line brute-force attacks. Particularly, with the recent developments of high-performance computing technologies such as graphics processing unit (GPU) technology and cloud computing technology, the probability of success for brute-force attacks increased dramatically. Researches show that the authentication four-way handshakes frames of WPA2-PSK (Pre-Shared Key) include hash value of secret key.[2] The attacker captures these frames and mounts off-line dictionary brute-force attack. Accelerated by GPU or

[1]School of Computer Science and Technology, Xi'an University of Posts and Telecommunications, Xi'an, China
[2]School of Information Science and Engineering, Central South University, Changsha, China

Corresponding author:
Qiong Zhang, School of Computer Science and Technology, Xi'an University of Posts and Telecommunications, Xi'an 710121, China.
Email: zhangqiong@xupt.edu.cn

cloud computing technologies, up to 130 million password combinations are ran through in just 20 min. Networks without strong password protection can be easily cracked.[3]

As attackers can capture wireless frames undetectably, and mount further off-line brute-force attack based on these frames, how to solve open channel problem attracted increasing attention in recent years. Algebraic channel decomposition multiplexing (ACDM)[4] is an approach to solve open channel problem through physical layer spread-spectrum precoding technology. In ACDM, the transmit code vectors are determined from the singular value decomposition (SVD) of the convolution matrix describing the channel between the transmitter and desired receiver. Since any potential transmitter–eavesdropper channel will have a different multipath structures, eavesdropper's ability to detect and decode the transmissions can be severely reduced. iJam[5] is another approach to solve open channel problem at physical layer. In iJam, the sender repeats its transmission two times. For each sample in these repeated transmissions, the receiver randomly jams either the sample in the original transmission, or the corresponding sample in the repetition. Since the eavesdropper does not know which signal sample is jammed and which one is clean, it cannot correctly decode the data.

ACDM and iJam solve open channel problem at physical layer, but such approaches require more complex wireless hardware devices. A great deal of time and efforts are needed to develop such devices. Furthermore, such approaches require the replacements of wireless devices that are widely used. Solving open channel problem without hardware modification is more attractive, which is the topic of this article.

It is widely accepted that cyclic redundancy check (CRC) checksum sequence is used in data link layer frame for communications, whether wired or wireless. CRC checksum is used for receivers to distinguish correct frames from error ones. Normally, correct frames are delivered to upper layers of communication protocols and error ones are dropped. In this article, we question this long-held design principle because the eavesdroppers also benefit from such facility. The eavesdroppers can capture correct frames for further brute-force dictionary attacks and drop the error ones.

If the checksum of frames are protected and only the intended receivers can decode it correctly, the eavesdroppers cannot distinguish correct frames from error ones. As frame error is inevitable in wireless networks, frames captured by eavesdroppers comprise both error frames and correct ones. Based on such error prone frames, results of the following off-line brute-force attacks are untrustworthy.

Therefore, checksum protection is an effective approach to solve open channel problem. As compared to approaches at physical layer, checksum protection at data link layer requires no hardware modification and can be implemented on widely used wireless devices. In this article, a reluctant checksum scheme named R-CS (reluctant checksum sequence) is proposed. R-CS protects checksum by improving ARQ (automatic repeat request) of media access control (MAC) sub-layer.

Traditional CRC frame checksum sequence is removed from DATA frame and two new checksums are used, cumulated checksum and hidden checksum. Cumulated checksum is the exclusive OR (XORed) result of CRC values of all frames successfully received by the intended receiver. Hidden checksum is encrypted by cumulated checksum, which replaces traditional CRC checksum sequence in the data frame.

The sender knows exactly the checksum of every frame. Therefore, it has the cumulated checksum by XOR all checksum of the transmitted frames. The receiver also maintains cumulated checksum by XOR checksum of all correctly received frames. If bad frames received, the receiver requests the sender to retransmit the frame until correct frame received. As a result, both the sender and the receiver know the cumulated checksum. But for the eavesdroppers, they cannot request retransmission for error frames received. A single error frame will lead to failure in tracing the cumulated checksum. All subsequent frames cannot be verified because lack of cumulated checksum. As wireless channel is prone to error, the eavesdroppers cannot capture all frames without error. R-CS can protect checksum to address open channel problem.

The contributions of this article are twofold. First, to the best of the authors' knowledge, this is the first time to argue that the widely used CRC frame checksum sequence is not reasonable from the perspective of wireless security. Using checksum explicitly in data link layer frame makes open channel problem more serious. It not only facilitates the receiver to process the frame effectively, but also benefit the eavesdroppers to capture correct frames for future attacks. Second, a reluctant checksum scheme R-CS is proposed to protect the checksum. Based on the inherent characteristic of wireless channel, R-CS prevents the eavesdroppers from accessing the checksum of frames. Therefore, the adversaries are under the risk of attacking the wireless networks based on error frames. The success probability of such attacks is decreased dramatically.

Adversary model and introduction of R-CS

Adversaries can be divided into two categories: passive and active. Passive adversaries eavesdrop on communications between terminals in the network. Active adversaries can forge frames with any addresses and any payload data. In this article, we assume that adversaries have both of the above capabilities: an adversary can capture all the communications in the network and forge frames with any data in the network.

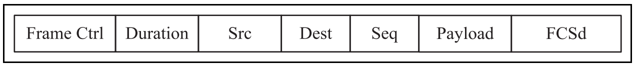


Figure 1. DATA frame at link layer.

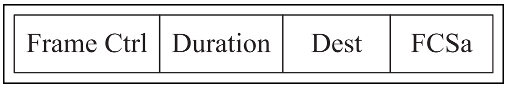


Figure 2. ACK frame.

As R-CS is based on improvement of ARQ scheme at data link layer, we give a brief introduction of ARQ first. To simplify presentation, denote the transmitter with *S*, and the receiver with *R*. Function *CRC(x)* is used to calculate CRC checksum of *x*. *DATAn* denotes the *n*th DATA frame from *S* to *R*, and *ACKn* denotes *n*th ACK frame from *R* to *S*.

In ARQ,

Step 1. *S* sends frame *DATAn* to *R*, field *Seq* is set to *n*. Structure of DATA frame is shown in Figure 1.

Frame Ctrl is used for frame control such as frame fragmentation. *Duration* is used for virtual carrier sensing. *Src* and *Dest* denote source address and destination address, respectively. *Seq* is the sequence number of this frame. *Payload* is the payload from upper layers. *FCSd* is the CRC checksum of this frame. That is, *FCSd = CRC(DATAn)*.

Step 2. On receiving *DATAn* (may be corrupted, denotes with *DATAn'*), *R* calculates the CRC checksum of *DATAn'* and compare it with *FCSd*. If *FCSd = CRC(DATAn')*, the frame is received correctly; otherwise error occurs. After receiving correct *DATAn, R* delivers *DATAn* to upper layers and responses to *S* by sending *ACKn* to *S*. DATA frame with error is discarded by *R* without any ACK to *S*. ACK frame is shown in Figure 2.

Step 3. *FCSa = CRC(ACKn)* is checksum of *ACKn* and *ACKn'* is ACK frame received by *S*. After sending *DATAn, S* starts a timer. If *ACKn* is received within TIMEOUT and *FCSa = CRC(ACKn'), S* continue to transmit *DATAn + 1*. Otherwise, if *ACKn* is not received within TIMEOUT or *ACKn* is received but *FCSa ≠ CRC(ACKn'), S* retransmits *DATAn*.

ARQ is efficient and adopted by almost all wireless communication technologies. But as explained above, CRC checksum sequence in DATA frame facilitates adversaries to eavesdrop correct frames and drop error ones. However, some modifications to ARQ can change this situation.

First, if *FCSd* is removed from *DATAn* and the checksum of it is appended to *ACKn, S* can still verify whether *R* received *DATAn* correctly. Second, if some steps are taken to protect the checksum in *ACKn*, the checksum is successfully protected from been eavesdropped. As the adversaries can only access DATA frames without checksum, they cannot distinguish error frames from correct ones.

R-CS is proposed based on above observations. The checksum of DATA frame is carried by ACK frame to *S*. And the CRC checksum in ACK frame is protected by accumulated checksum. Frames not successfully received by *R* will be retransmitted. Therefore, *S* and *R* know exactly the frames that already have been accepted by *R*. Traditional CRC can be used to calculate checksum of these frames. With these checksums, the cumulated checksum can be calculated by simple XOR all of them, which is used for subsequent checksum protection. Cumulated checksum is updated after successful frame transmission. Apparently, cumulated checksum is dynamic and shared by *S* and *R*. Adversaries can trace cumulated checksum only if all frames are captured correctly; a single error frame will make it failed to trace the cumulated checksum. Without cumulated checksum, adversaries lose checksum of all the subsequent frames. As wireless channel is error prone and adversaries cannot request *S* for retransmission, it is impossible for them to capture all frames without error for a long time. Without frame checksum, the adversaries cannot distinguish error frames from correct ones. It is quite difficult to mount success attacks based on such frames.

Detailed description of R-CS

Algorithm 1: basic R-CS

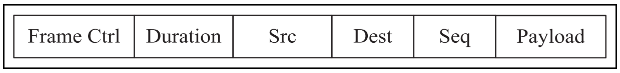
Step 1. *S* maintains two local variables, *Hsc* and *Hsh*. *Hsc* is the cumulated checksum, and *Hsh* is the hidden checksum. Initially, *Hsc* = 0 × 0. *S* calculates the hidden checksum of *DATAn* first

$$Hsh = h_1(Hsc, DATAn) \tag{1}$$

And then *S* updates the cumulated checksum

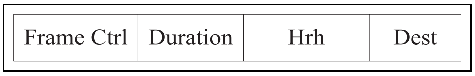
$$Hsc = h_2(DATAn) \oplus Hsc \tag{2}$$

As compared to normal wireless communications, FCS is removed from DATA frame in R-CS, which is illustrated in Figure 3.



Frame Ctrl	Duration	Src	Dest	Seq	Payload

Figure 3. DATA frame structure in R-CS.



Frame Ctrl	Duration	Hrh	Dest

Figure 4. ACK frame structure in R-CS.

After the checksum calculations and local variable updates, S transmits $DATAn$.

Step 2. R maintains three variables, Hrc, Hrh, and $Hrtmp$. Hrc is the cumulated checksum, Hrh is hidden checksum, and $Hrtmp$ is the temporary checksum. The initial value of Hrc is also set to 0×0. On receiving $DATAn'$, R calculates temporary checksum of $DATAn'$

$$Hrtmp = h_2(DATAn') \tag{3}$$

And the hidden checksum of $DATAn'$ is also calculated

$$Hrh = h_1(Hrc, DATAn') \tag{4}$$

And then, R sends $ACKn$ to S. As illustrated in Figure 4, the $ACKn$ includes hidden checksum of the corresponding $DATAn$.

Step 3. On receiving $ACKn$, S compares local hidden checksum Hsh and received hidden checksum Hrh. If $Hsh = Hrh$, R has received $DATAn$ correctly. S continues to transmit next frame, $DATAn + 1$. Otherwise, if $Hsh \neq Hrh$, error occur and $DATAn$ must be retransmitted. Note that the error is introduced either by incorrect $DATAn$ received by R or error in $ACKn$. Both situations are deemed as error by S. No local variables update is necessary for retransmission.

Step 4. On receiving $DATAn$, R checks Seq field to determine whether the received frame is $DATAn$ or $DATAn + 1$ first. If $Seq = n$, this is a retransmission of $DATAn$, turn to Step 2. Otherwise, if $Seq = n + 1$, no error happens during $DATAn$ and $ACKn$ transmission. R updates the local variables

$$Hrc = Hrtmp \oplus Hrc \tag{5}$$

And continue to Step 2.

The detailed algorithm can be formally described as follows:

```
  Sender:
  $Hsc \leftarrow 0$ x $0$;
  $Seq = 0$;
  On have packet to be sent:
    $Hsh \leftarrow h_1(Hsc||DATAn)$;
    $Hsc \leftarrow h_2(DATAn) \oplus Hsc$
    Send DATAn without checksum;
    Start timer;
  On receive ACK frame:
    if $Hsh = Hrh$ then
      $Seq++$; //next frame
    else
      Retransmit DATAn;
    end;
  On timeout:
      Retransmit DATAn;
Receiver:
  $Hrc \leftarrow 0$ x 0;
  $Seq \leftarrow 0$;
  On receive DATAn:
    if $n = Seq + 1$ then //transmit successfully
      $Hrc \leftarrow Hrtmp \oplus Hrc$
      $Hrtmp \leftarrow h_2(DATAn + 1')$
      $Hrh \leftarrow h_1(Hrc, DATAn + 1')$
      Send ACK with Hrh;
      Seq++;
    else if $n = Seq$ then
      $Hrtmp \leftarrow h_2(DATAn')$
      $Hrh \leftarrow h_1(Hrc, DATAn')$
      Send ACK with Hrh;
    end if;
```

As compared to traditional approaches, the checksum of DATA frame is acknowledged to S in ACK frame, instead of being carried in DATA frame. The adversaries cannot get the checksum easily by capturing DATA frame. Of course, moving the checksum from DATA frame to ACK frame is not a challenge to adversaries because they get the checksum by capturing both the DATA frame and subsequent ACK frame. The key step for checksum protection is the introduction of cumulated checksum and hidden checksum. Cumulated checksums Hsc and Hrc are the XORed results of all DATA frames that have been successfully received by R. Retransmissions for error frames keep Hsc and Hrc synchronized for S and R. Hrh and Hsh are called hidden checksum. Hidden checksum is the encrypted hash value of new DATA frame. During normal operations, R transmits Hrh in ACK frame for checksum verify purpose. As Hsc and Hrc are synchronized for R and S, the verify process is easy. But to the adversary, to confirm that the captured DATA frame is correct, it needs correct cumulated checksum. That means the adversary must capture all previous DATA frames without error. Assuming that *DATAn* is not captured correctly and all previous DATA frame are correct, adversary can confirm that *DATAn* is corrupted by calculating the hidden checksum in equation (4) and comparing it with *Hrh* in the subsequent ACK frame. DATA frames can be dropped by the adversary because of frame error. But after that, the adversary cannot distinguish error frames from correct ones because of lost synchronization of cumulative checksum.

Sequence number protection in R-CS

R-CS protects checksum by improving ARQ. But researches in wireless security revealed that ARQ is prone to packet injection attacks.[6,7] Therefore, R-CS is also under the risk of packet injection attacks, and precautions must be taken. Sequence number is used to identify DATA frames. *Seq* field of DATA frame is used for this purpose. *R* uses *Seq* to determine whether this is a required frame. Assuming that previous received DATA frame is *DATAn*, the next DATA frame must be $DATAn + 1$. All DATA frames with *Seq* field not equal to $n + 1$ are discarded. To launch a packet injection attack against R-CS, the adversary can forge a packet with $Seq = n + 1.R$ receives it unaware of the attack, and acknowledges with $ACKn + 1$. The adversary can further forge $DATAn + 2$. According to *algorithm1, R* considers that $DATAn + 1$ is correct and delivers it to upper layer. R-CS is cracked because the following DATAn from *S* will never be accepted by *R*.

R-CS adopts sequence number hashing to fight against such attacks. As *Hsc* and *Hrc* are known to *S* and *R*, respectively, and adversary knows nothing about them, they can be used to protect sequence number against attacks. For this purpose, some modifications to DATA frame are necessary. A field name IV is appended to DATA frame. IV is a random vector to introduce computation complexity to adversaries. *Seq* field of DATA frame is set to $h(Hsc||Seq||IV)$ by *S*. As $Hrc = Hsc$ and the next sequence number are known, *R* can calculate the hash value easily. If the result is equal to *Seq* in DATA frame, the sequence number is correct; otherwise, error frame received or attacks occur. But to the adversaries, as cumulated checksum is unknown, they cannot forge DATA frames any more.

The last frame problem and the solution

In R-CS, assuming that *S* sends a DATA frame to *R*, and no more frames are to be transmitted, the last frame problem occurs. If *S* sends *DATAn* to *R, R* acknowledges *S* with *ACKn*. *DATAn* cannot be delivered to upper layer in R-CS. If no more frames from *S, DATAn* keeps undelivered. This will cause serious delay to *DATAn*.

The following steps can be taken to solve the last frame problem. On receiving *ACKn* from *R, S* verifies checksum as described in *algorithm1*. If the checksum is correct and no more frames available for transmission, *S* sends a random DATA frame with sequence number $n + 1$. On receiving $DATAn + 1$, *R* delivers *DATAn* to upper layer. Having more frames to be transmitted, *S* transmits a real frame with $Seq = n + 1$. *R* treats previously received $DATAn + 1$ as error frame and discarded.

Protecting key frames

R-CS improves wireless security through protecting checksum of DATA frames. To prevent adversaries from eavesdropping key frames (such as four-way handshakes of WPA2), R-CS uses active fake frame injection. As sequence numbers of frame are protected in R-CS, the adversaries cannot distinguish retransmission from normal transmission. Therefore, *S* can insert random frames to cheat adversaries. For example, *S* sends *DATAn* with random payload;*R* acknowledges it with *ACKn*. But *S* can force *R* to discard previous DATA frame by send another *DATAn*, even if it is correctly received by *R*. By the introduction of active fake frame injection, *S* can inject random frames without confusing *R*. But to the adversaries, fake frames are hard to filter. Useless frames eavesdropped by the adversaries poses more challenge to adversaries and make it almost impossible to launch success brute-force attacks based on such frames.

Selection of the checksum functions

Two different checksum functions are adopted in R-CS, $h_1(\cdot)$ and $h_2(\cdot)$. $h_2(\cdot)$ is used to calculate the checksum of DATA frame, and $h_1(\cdot)$ is used to encrypt checksum of DATA frame. Selection of the checksum functions is a tradeoff between performance and security level. As hidden checksum is transmitted in ACK frame, it can be captured by adversary and cracked through brute-force attacks. Therefore, selection of $h_1(\cdot)$ and its parameters are focused on security. $h_1(\cdot)$ takes two parameters, the cumulated checksum and the DATA frame. Message authentication functions can be used for $h_1(\cdot)$. Taking the cumulated checksum as the key, the resulting message authentication code can be used for integration check. For performance consideration, efficient MAC functions such as Universal MAC (UMAC)[8] can be used. According to the characteristics of MAC function, the computation complexity of brute-force attacks on MAC key is 2^k, where k is the length of the key. Long key improves security. From equations (1) and (2), MAC key in R-CS is the output of $h_2(\cdot)$. Therefore, one-way hash functions with low computation complexity can such as Tiger,[9] CRC64,[10] and MD5 can be used.

Simulation results

R-CS prevents checksum from being eavesdropped. But it may be worried that the received signal strength (RSS) may be used for indication of frame error. Apparently, it is easy for adversaries to get RSS for DATA frames. If packet error can be deduced via RSS, R-CS will be bypassed, and the adversaries can drop error frames without checksum. As we known, the main reasons for frame error are weak signal and collisions.[11,12] First, the error frames introduced by collisions are irrelevant to RSS. Frames may be corrupted because of collision even if the RSS is pretty strong.[14] And moreover, previous researches show that carrier sense media access (CSMA)/collision avoid (CA)-based MAC protocols such as 802.11 cannot eliminate hidden terminal problem. Collisions introduced by hidden terminal problem are the main source for frames error in high densely deployed wireless network. Second, there are no established relations between RSS and error frame even if frame error is introduced by weak signals. Without checksum, the adversaries cannot confirm that the frame captured is correct only by RSS. To validate the irrelevance between RSS and frame error, the first simulations are conducted.

GloMoSim[13] is used for the simulations. As shown in Figure 5, there are 10 nodes deployed in the network. Constant bit rate (CBR) traffic model is used. Node 0 sends data to node 9 at 200 kbps in a multi-hop fashion. The build-in CSMA/CA MAC scheme of GloMoSim is used for the simulation. By adjusting the distance between nodes, several simulations are conducted. The results are shown in Figure 6.

Simulation results show that frame error cannot be avoided even if nodes have very strong RSS. The simulations are configured for three payload size, which means different frame sizes. Short frame results in low packet error rate, but cannot eliminate it. When the RSS is pretty strong (such as −40 dBm), the frame error rate remains not less than 2.1% because of collisions. Simulations with more nodes in the network are carried out, and the results reveal that more nodes in the network lead to greater frame error rate. Therefore, error frames are inevitable, even if the RSS is fairly strong.

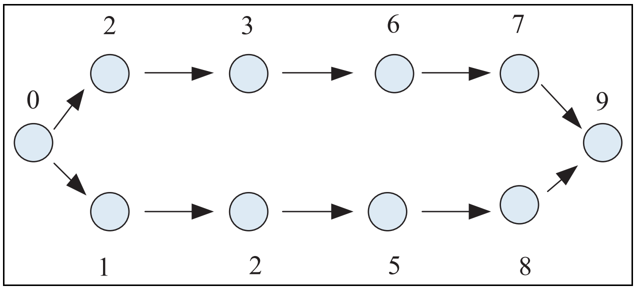


Figure 5. Simulation topology.

R-CS is implemented in GloMoSim. As the adversaries lost cumulated checksum if a single frame is corrupted, the time from the beginning of the transmission to the first error frame (we call lose sync time) is an important security metric for R-CS. During lose sync time, the adversaries can trace checksum by ACKs. Simulations are conducted to measure lose sync time in the topology used in previous simulation, except that an adversary node is deployed. The adversary is place much close to node 9 to capture DATA frames to it. The results are shown in Figure 7.

Simulation results show that lose sync time is less than 1.2 s irrespective of the frame size. Further analyses on the trace log of the simulation show that frame error is mainly introduced by frame collisions. As the adversary is place much close to the sender, there are less error frames introduced by weak signals. R-CS can prevent adversaries from accessing the checksum and solve the open channel problem.

Summary and conclusion

R-CS solves open channel problem at data link layer, which implies that no hardware modification is required. This is very important as wireless devices are widely deployed. Only checksum calculation is required to implement R-CS. Low computation complexity and

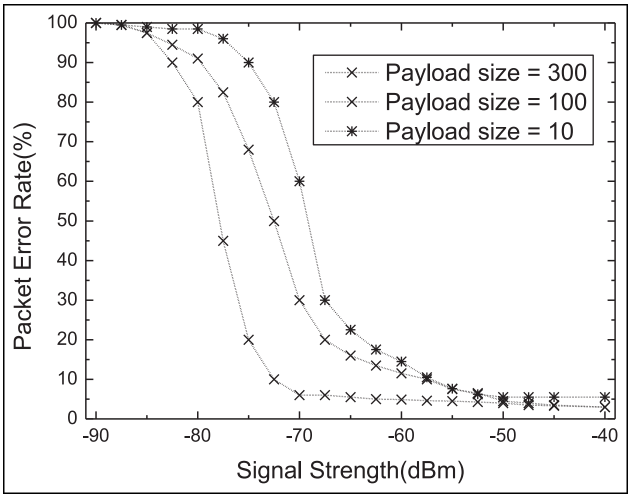


Figure 6. Signal strength and packet error rate (PER).

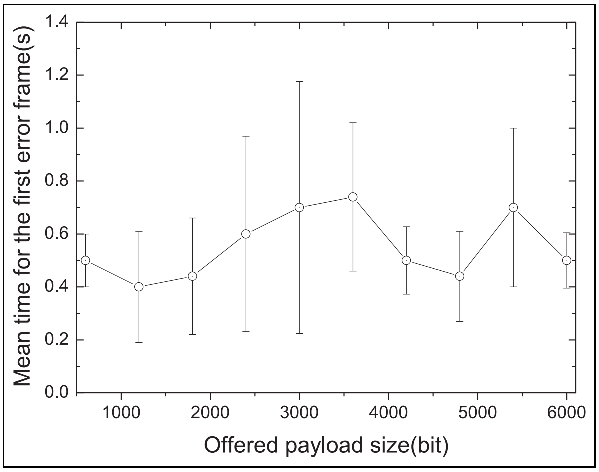


Figure 7. Mean time for first error frame.

easy implementation make it particularly suitable for wireless sensor networks.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article.

References

1. Mathur S, Reznik A, Ye C, et al. Exploiting the physical layer for enhanced security [security and privacy in emerging wireless networks]. *IEEE Wirel Commun* 2010; 17(5): 63–70.
2. Gold S. Cracking wireless networks. *Netw Secur* 2011; 2011(11): 14–18.
3. Li S, Da Xu L and Zhao S. The internet of things: a survey. *Inform Syst Front* 2015; 17: 243–259.
4. Sperandio C and Flikkema PG. Wireless physical-layer security via transmit precoding over dispersive channels: optimum linear eavesdropping. In: *Proceedings of MILCOM 2002*, Anaheim, CA, 7–10 October 2002, pp.1113–1117. New York: IEEE.
5. Gollakota S and Katabi D. Physical layer wireless security made fast and channel independent. In: *Proceedings of IEEE INFOCOM 2011*, Shanghai, China, 10–15 April 2011, pp.1125–1133. New York: IEEE.
6. Kim KH. Security attack based on control packet vulnerability in cooperative wireless networks. In: *The ninth international conference on networking and services*, Lisbon, 24–29 March 2013, pp.123–128. Lisbon, Portugal: IARIA.
7. Li S, Da Xu L and Wang X. On ARQ-based wireless communication systems in the presence of a strategic jammer. In: *IEEE global conference on signal and information processing*, 2014, pp.478–483. Georgia, USA: IEEE.
8. Black J, Halevi S, Krawczyk H, et al. UMAC: fast and secure message authentication. In: *Proceedings of the 19th annual international cryptology conference on advances in cryptology*, Santa Barbara, CA, 15–19 August 1999, pp.216–233. London: Springer-Verlag.
9. Anderson RJ and Biham E. Tiger: a fast new hash function. In: *Proceedings of the third international workshop on fast software encryption*, Cambridge, 21–23 February 1996, pp.89–97. London: Springer-Verlag.
10. Hlávka P, Kratochvíla T, Rehak V, et al. *CRC64 algorithm analysis and verification 1*. Prague, Czech Republic: CESNET technical report, December 2005.
11. Rodenas-Herraiz D, Garcia-Sanchez AJ, Garcia-Sanchez F, et al. On the improvement of wireless mesh sensor network performance under hidden terminal problems. *Future Gener Comp Sy* 2015; 45: 95–113.
12. Li S and Da Xu L. *Securing the internet of things*. 1st ed. Cambridge, MA: Elsevier, 2017.
13. Zeng X, Bagrodia R and Gerla M. GloMoSim: a library for parallel simulation of large-scale wireless networks. In: *Proceedings of the twelfth workshop on parallel and distributed simulation*, Banff, AB, Canada, 29 May 1998, pp.154–161. New York: IEEE.
14. Rayanchu S, Mishra A, Agrawal D, et al. Diagnosing wireless packet losses in 802.11: separating collision from weak signal. In: *IEEE 27th conference on computer communications*, Phoenix, AZ, 13–18 April 2008, pp.735–743. New York: IEEE.