AN TOÀN DỮ LIỆU TRONG MẠNG CẢM BIẾN KHÔNG DÂY

Lê Hoàng Anh[1]

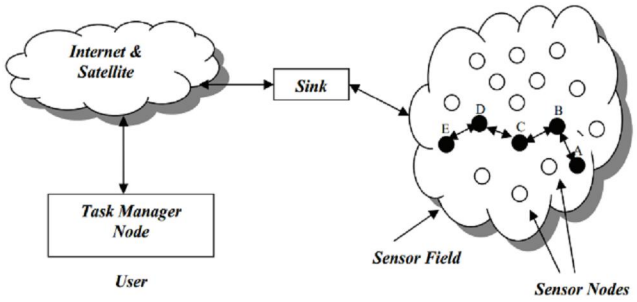
TÓM TẮT

Mạng cảm biến không dây có nhiều ưu điểm và đang được áp dụng phát triển mạnh mẽ. Tuy nhiên mạng vẫn còn nhiều thách thức cần được giải quyết như: nguồn năng lượng bị giới hạn, các nút (node) cảm biến có bộ nhớ và bộ vi xử lý thấp, đặc biệt là các cơ chế an ninh còn hạn chế. Việc bảo mật trong mạng là vấn đề đầy thách thức và đang được các nhà khoa học tiếp tục nghiên cứu. Bài viết này sẽ trình bày những lợi ích của mạng cảm biến không dây và lý giải những vấn đề bảo mật vẫn còn nhiều thách thức. Từ đó bài viết sẽ trình bày các thuật toán mã hóa và xác thực có thể áp dụng trong mạng cảm biến không dây. Chúng tôi đã cài đặt, thực nghiệm và đưa ra hai cơ chế bảo mật trong mạng cảm biến không dây.

Từ khóa: *An toàn dữ liệu, bảo mật, cảm biến, mạng cảm biến không dây.*

1. ĐẶT VẤN ĐỀ

Mạng cảm biến không dây là mạng lưới các thiết bị có kích thước nhỏ, nhiều nút cảm biến, các nút cảm biến có thể cảm nhận môi trường và truyền thông tin thu thập được từ các vùng giám sát về sink thông qua kết nối không dây. Mạng cảm biến không dây được ứng dụng trong các lĩnh vực như: an ninh quốc gia, giám sát, quân sự, chăm sóc sức khoẻ, giám sát môi trường và nhiều lĩnh vực khác [9].



Hình 1. Kiến trúc truyền thông mạng cảm biến không dây [9]

Tuy mạng cảm biến không dây có nhiều ứng dụng trong cuộc sống, nhưng mạng cũng có những khó khăn và hạn chế gặp phải khi triển khai như: giới hạn về năng lượng, giới hạn về băng thông, giới hạn về phần cứng, giới hạn về kết nối. Trong đó thách thức và trở ngại lớn nhất là nguồn năng lượng bị giới hạn không thể nạp lại, vấn đề bảo mật và an ninh mạng. Trong mạng cảm biến không dây, năng lượng được sử dụng chủ yếu cho 3 mục đích: truyền dữ liệu, xử lý dữ liệu và đảm bảo cho phần cứng hoạt động. Hiện nay, các nhà khoa học đang nghiên cứu phát triển mạng cảm biến không dây vừa đảm bảo về mặt bảo mật dữ liệu nhưng cũng phải đảm bảo các yêu cầu về năng lượng.

[1] *Khoa Công nghệ thông tin, Trường Đại học An Giang*

Trong mạng cảm biến không dây, các gói tin dễ bị tấn công theo các cách khác nhau như: tấn công từ chối dịch vụ, tấn công thông tin quá cảnh, tấn công Sybil, tấn công Blackhole/Sinkhole, tấn công Hello Flood, tấn công Wormhole [1]. Từ khảo sát các cuộc tấn công và các mối đe dọa đối với mạng cảm biến không dây các nhà nghiên cứu đã đưa ra các chương trình bảo mật khác nhau cho mạng cảm biến không dây như: JAM; Wormhole based; Statistical En-Route Filtering; Radio Resource Testing, Random Key Pre-distribution; Bidirectional Verification, Multi-path multi-base station routing; On Communication Security; TIK; Random Key Predistribution; REWARD; SNEP & µTESLA [1].

Thông thường trong các mạng cảm biến không dây, tính xác thực, tính toàn vẹn, tính bảo mật thông điệp thường được thực hiện bởi cơ chế bảo mật đầu cuối-đầu cuối như SSH, SSL, TLS hoặc IPSec; các bộ định tuyến trung gian chỉ cần xem tiêu đề của thông điệp rồi chuyển tiếp mà không cần phải xem nội dung của thông điệp. Cơ chế bảo mật đầu cuối đến đầu cuối dễ bị tấn công [7]. Nếu tính toàn vẹn thông điệp chỉ kiểm tra tại điểm cuối cùng, mạng có thể định tuyến chuyển các gói tin đã bị tấn công qua nhiều *hop* trước khi chúng bị phát hiện. Loại tấn công này sẽ làm lãng phí năng lượng và băng thông, đây được xem là tài nguyên quý giá trong mạng cảm biến không dây. Kiến trúc bảo mật lớp liên kết dữ liệu có thể phát hiện các gói dữ liệu bất hợp pháp khi lần đầu tiên chúng được đưa vào mạng. Cơ chế bảo mật lớp liên kết dữ liệu còn được đề xuất để chống lại các cuộc tấn công từ chối dịch vụ.

Với những lý do đó, chúng tôi quyết định chọn cơ chế kiến trúc bảo mật trên lớp liên kết dữ liệu trong mạng cảm biến không dây. Cơ chế bảo mật trên lớp liên kết dữ liệu đảm bảo tính xác thực, tính toàn vẹn, bảo mật các thông điệp và trên lớp liên kết dữ liệu có thể tối ưu kích thước các gói tin khi truyền, từ đó giảm băng thông và năng lượng các nút, trên lớp liên kết dữ liệu có thể kiểm tra tính xác thực của gói tin ngay khi gói tin được đưa vào mạng nên có thể loại bỏ ngay khi các gói tin bất hợp pháp được đưa vào mạng mà không phải truyền qua nút khác từ đó làm giảm việc tiêu hao năng lượng của các nút khi truyền các gói tin bất hợp pháp.

Tiếp theo bài viết sẽ trình bày: phân tích các thuật toán mã hóa và xác thực có thể áp dụng trong mạng cảm biến không dây, đánh giá kết quả bảo mật của các thuật toán trong mạng cảm biến không dây, thực nghiệm đưa ra các cơ chế bảo mật, phân tích ưu nhược điểm tương ứng với mỗi cơ chế.

2. NỘI DUNG NGHIÊN CỨU

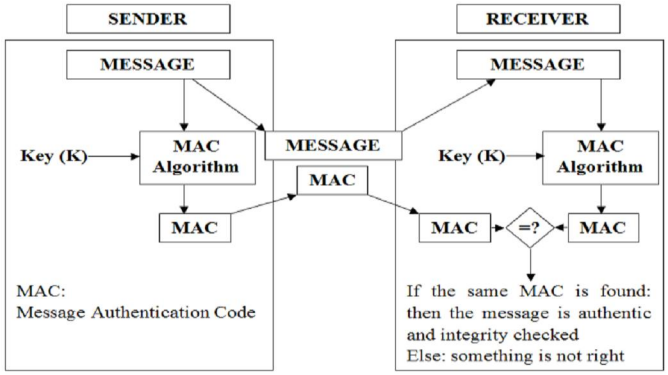
2.1. Phương pháp nghiên cứu

Bảo mật dữ liệu trên lớp liên kết dữ liệu nhằm đảm bảo tính toàn vẹn, tính xác thực, tính bảo mật.

2.1.1. Tính toàn vẹn và tính xác thực

Tính toàn vẹn và xác thực dữ liệu trong mạng có thể đạt được thông qua mã xác thực thông điệp MAC (Message Authentication Codes) được gọi là hàm băm có khóa, đầu vào là một khóa bí mật và dữ liệu để được xác thực và đầu ra là giá trị MAC. Giá trị MAC đảm

bảo tính toàn vẹn và tính xác thực của dữ liệu bằng cách so sánh giá trị MAC để phát hiện sự thay đổi nội dung của dữ liệu [11].



Hình 2. Mô hình hoạt động MAC giữa bên gửi và bên nhận

Bên gửi sẽ tính toán giá trị MAC dựa vào thông điệp và khóa K, giá trị MAC sẽ được gửi cùng với thông điệp. Bên nhận sẽ tính toán lại giá trị MAC và so sánh với giá trị MAC trong thông điệp, nếu giống thì thông điệp được xác thực ngược lại thông điệp bị loại bỏ [11].

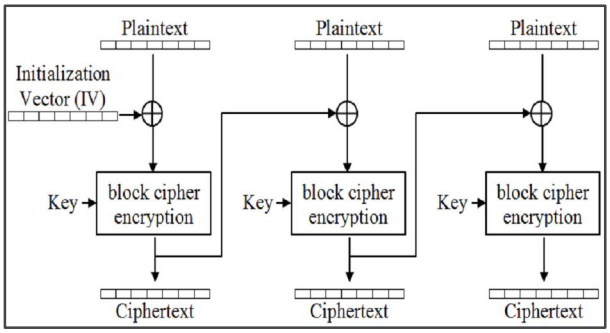
Hàm MAC có khả năng chống giả mạo các bản rõ của các cuộc tấn công mạng. Giá trị MAC được tạo ra và được xác thực cùng một khóa bí mật (hình 2).

Các thuật toán MAC được xây dựng dựa trên các mật mã nguyên thủy khác như hàm băm mật mã (như trong trường hợp của HMAC) hoặc từ các thuật toán mã hóa khối (OMAC, CBC-MAC và PMAC) [11].

2.1.2. Tính bảo mật

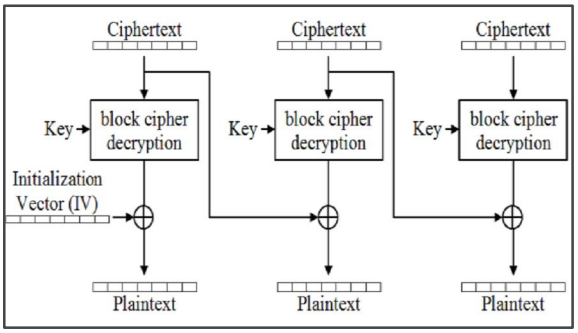
Tính bảo mật trong mạng có thể đạt được thông qua thuật toán mã hóa Skipjack, là một thuật toán mã hóa cho việc truyền tải thông tin được phát triển bởi cơ quan an ninh quốc gia Mỹ. Skipjack sử dụng các thuật toán trao đổi khóa Diffie-Hellman cho việc phân phối các khóa phiên [10].

Skipjack sử dụng một khóa 80 bits để mã hóa hoặc giải mã các khối dữ liệu 64 bits. Skipjack sử dụng một mạng Feistel không cân bằng với 32 vòng để mã hóa hoặc giải mã [6].



Hình 3. Sơ đồ mã hóa khối chuỗi [3]

Theo hình 3, thông điệp được chia thành các khối theo kích thước cố định, khối đầu tiên được mã hóa với vector khởi tạo và khóa để tạo ra mật mã và mật mã này tiếp tục làm tham số đầu vào cho khối tiếp theo, quy trình như vậy được lặp lại cho đến khối cuối cùng sẽ tạo ra được các khối mật mã.



Hình 4. Sơ đồ giải mã khối chuỗi [3]

Theo hình 4, khối mật mã đầu tiên sẽ được giải mã với tham số đầu vào là khóa và vector khởi tạo để giải mã, khối mật mã đó tiếp tục làm tham số đầu vào cùng với khóa cho khối mật mã tiếp theo để giải mã, quá trình tiếp tục đến khối mật mã cuối cùng để giải mã thành các bản rõ ban đầu.

Skipjack có thể được sử dụng cùng với vector khởi tạo (Initialization Vector - IV) như một tham số đầu vào cùng với khóa bí mật *K* nhằm làm tăng tính bảo mật của dữ liệu được mã hóa. Các chương trình mã hóa chủ yếu sử dụng giá trị *IV* được tạo ngẫu nhiên để đảm bảo an toàn ngữ nghĩa, do tính chất này mà nhờ đó việc sử dụng lặp đi lặp lại của chương trình cùng 1 dữ liệu với cùng 1 khóa ngăn không cho phép kẻ tấn công suy ra mối quan hệ giữa các phân đoạn của thông điệp được mã hóa [6].

Kích thước của *IV* phụ thuộc vào mật mã gốc được sử dụng, đối với mật mã khối thì kích thước của *IV* thường là kích thước khối của thuật toán mã hóa. Khi chọn kích thước cho *IV* phải tính tới xác suất đụng độ do vấn đề lặp lại *IV* phải được tính toán và cân nhắc [6].

2.1.3. Đánh giá việc bảo mật dữ liệu trên lớp liên kết dữ liệu

2.1.3.1. Tính bảo mật thông điệp

Để sử dụng mã hóa ngữ nghĩa an toàn đòi hỏi phải có chương trình mã hóa và xác định định dạng *IV* phù hợp.

Mã hóa khóa đối xứng thường có 2 loại: mật mã dòng và phương thức hoạt động sử dụng mật mã khối. Mật mã dòng thường sử dụng một khoá *K* và vector khởi tạo *IV* như tham số đầu vào của hàm *pseudorandom keystream* GK(IV). *Keystream* sau đó được XOR với thông điệp như công thức (1) để có được bản mã:

$$C = (IV, G_K (IV) \oplus P) \quad (1)$$

Theo một số tài liệu thì mật mã dòng nhanh hơn mật mã khối trong những môi trường xử lý hạn chế về tài nguyên [8]. Nhưng bất lợi của mật mã dòng là nếu dùng cùng *IV* để mã hóa 2 gói tin khác nhau thì có thể phục hồi lại được cả 2 bản rõ (plaintext). Ví dụ cho C = (IV, GK (IV) ⊕ P) và C' = (IV, GK (IV) ⊕ P'), ta có thể phục hồi rất nhiều thông tin của P và P' từ P ⊕ P', thường thì có thể phục hồi hầu hết thông tin của P và P' từ P ⊕ P'. Để đảm bảo rằng *IV* không lặp lại đòi hỏi *IV* phải khá dài ít nhất 8 bytes. Như đã nói ở trên trong điều kiện hạn chế về nguồn tài nguyên của mạng cảm biến không dây chúng ta phải hạn chế chi phí gói tin càng ít càng tốt. Trong ngữ cảnh này phải thêm 8 bytes trong một gói tin 30 bytes nghĩa là chiếm tới gần 27% (hơn 1/4) tổng gói tin thì khó có thể chấp nhận được và nếu *IV* ngắn hơn thì việc lặp lại *IV* sẽ xảy ra và điều này không đảm bảo về mặt bảo mật nên trong bài này chúng tôi không sử dụng mật mã dòng mà thay vào đó sẽ là mật mã khối.

Thuật toán mã hóa khối là một *hàm giả ngẫu nhiên* có khóa trên các chuỗi bit nhỏ, thường là 8 hoặc 16 bytes. Các thuật toán thuộc họ mã khối bao gồm DES, AES, RC5 và Skipjack. Để mã hóa những thông điệp dài hơn 8 hoặc 16 bytes, mật mã khối chia thông điệp thành từng khối nhỏ để mã hóa. Ví dụ một mật mã khối *k* byte thì nó sẽ ngắt thông điệp thành những đoạn *k* byte và thuật toán mã hóa khối sẽ mã hóa thông điệp theo từng khối. Ngoài ra khi sử dụng thuật toán mã hóa khối còn có một số thuận lợi: thuật toán mã xác thực thông điệp sẽ hoạt động hiệu quả hơn trên mã hóa khối, sử dụng thuật toán mã hóa khối sẽ không làm tăng chiều dài mật mã sau khi mã hóa. Để sử dụng thuật toán mã

hóa khối để mã hóa thì phải chọn một phương thức (chế độ) hoạt động phù hợp. Thông thường là sử dụng chế độ *counter* (CTR).

Khi nói tới thuật toán mã hóa khối thì người ta thường nghĩ tới một trong hai thuật toán AES hoặc Triple-DES. Tuy nhiên, AES và Triple-DES thì quá chậm để thực hiện trong các vi điều khiển nhúng [2] vì thế thuật toán AES và Triple-DES sẽ loại bỏ không được sử dụng ở bài viết này. Do đó với những phân tích ở trên thì thuật toán Skipjack sẽ thích hợp hơn trong mạng cảm biến không dây nên chúng tôi sẽ chọn thuật toán Skipjack là thuật toán mã hóa trong bài viết này.

Mục tiêu của chúng tôi là bảo mật nhưng phải làm sao có thể giảm chi phí cho việc bảo mật càng ít càng tốt. Chiều dài của *IV* và cách tạo ra *IV* có thể ảnh hưởng lớn đến việc bảo mật và hiệu suất. Nếu *IV* quá dài, sẽ dẫn đến thêm các bit không cần thiết vào các gói tin và có ảnh hưởng đáng kể đến chi phí về băng thông và tiêu hao năng lượng. Đồng thời, nếu *IV* quá ngắn thì *IV* có nguy cơ sẽ bị lặp lại và dẫn đến sự bảo mật trong mạng có thể sẽ không được đảm bảo.

Vậy *IV* dài bao nhiêu bit là đủ? Theo nguyên tắc chuồng bồ câu, một *IV* có n-bit sẽ lặp lại sau khi 2^n +1 gói tin được gửi đi. Nếu sử dụng *counter* n-bit thì việc lặp lại sẽ không xảy ra trước thời điểm 2^n + 1 gói tin được gửi. Tuy nhiên, với một số cách tạo ra *IV*, việc lặp lại có thể xảy ra trước đó. Nếu chọn mỗi *IV* như một giá trị n-bit ngẫu nhiên, sau đó sử dụng hàm *paradox* để tạo ra *IV* thì xác suất sự lặp lại *IV* đầu tiên là sau khoảng $2^{n/2}$ gói tin được gửi đi. Vì thế, chúng tôi sử dụng một *counter* trong *IV* và truyền tải nó trong các gói tin để bên nhận có thể biết được giá trị của *counter*.

Cấu trúc của *IV* là *Dst* || *AM* || *L* || *Src* || *Ctr*, Dst là địa chỉ đích của người nhận, AM là loại thông điệp, L là chiều dài payload của dữ liệu, Src là địa chỉ của người gửi và Ctr là một *counter* 16 bits. *Counter* bắt đầu từ 0 và bên gửi tăng nó lên 1 sau mỗi thông điệp được gửi đi.

Với 2 bytes *counter* như vậy chúng tôi muốn tối đa hóa số lượng *IV* cho mỗi nút và như vậy theo tính toán thì mỗi nút có thể gửi ít nhất 2^{16} gói tin trước khi *IV* lặp lại và với 1 mạng cảm biến không dây gồm n nút thì tổng số gói tin được gửi là n*2^{16} gói tin được gửi trước khi *IV* lặp lại. Đối với các mạng thông thường băng thông khoảng trên 1Mb/s thì thời gian để gửi 2^{16} gói tin là rất ngắn. Tuy nhiên với điều kiện hạn chế của mạng cảm biến không dây thì thời gian để gửi 2^{16} gói tin sẽ lớn hơn nhiều so với mạng thông thường. Ví dụ một ứng dụng mạng cảm biến không dây thực tế ở Great Duck Island các nút cảm biến sẽ gửi thông tin cảm biến được cứ mỗi 70 giây 1 lần. Như vậy có thể tính được thời gian để gửi 2^{16} gói tin với 70 giây sẽ gửi 1 lần là khoảng 53 ngày. Do đó thời gian này có thể được xem là hợp lý với mạng cảm biến không dây vì với các nút cảm biến mica2 thời gian sống khoảng 2 tuần. Đối với một số mạng nếu có thời gian sống trên 53 ngày thì việc lặp lại *IV* xảy ra chỉ gặp vấn đề về bảo mật an toàn khi dùng chung 1 khóa. Cách giải quyết khi *IV* lặp lại là khi gần đến thời gian lặp lại *IV* thì sẽ cập nhật lại khóa mới cho mạng cảm biến không dây.

Theo như cấu trúc của *IV* thì chúng ta còn 4 bytes *Dest||AM||L* để đảm bảo an toàn khi *counter* lặp lại. *IV* bao gồm cả phần *Dest||AM||L* điều này có nghĩa là nếu *counter* có lặp lại thì cũng chưa chắc *IV* cũng bị lặp lại. Như vậy, khi giá trị *counter* lặp lại thì thông tin chỉ bị lộ khi gửi 2 thông điệp cùng đến 1 địa chỉ Dest, cùng 1 loại AM, cùng chiều dài L và điều này rất ít khi xảy ra cũng như phải biết 2 thông điệp hoặc 2 gói tin nào được gửi cùng 1 *IV* mới có thể giải mã ra được bản rõ. Tóm lại với định dạng *IV* là *Dst||AM||L||Src||Ctr* thì có thể nói dữ liệu được bảo mật kép vì muốn giải mã được phải biết *IV* hoặc tìm được 2 thông điệp được mã hóa với cùng 1 *IV* (nghĩa là cùng 1 *counter*, cùng 1 Dest, cùng 1 AM và cùng 1 L) thì mới có thể giải mã được thông điệp.

2.1.3.2. Tính toàn vẹn và tính xác thực

Nếu chỉ có mã hóa không thì chưa đủ để bảo mật và thực tế đã chỉ ra rằng nếu sử dụng mã hóa mà không có xác thực thì sẽ không đảm bảo an toàn và vì thế ngoài việc mã hóa chúng ta cần có cơ chế xác thực thông điệp. Tính toàn vẹn và tính xác thực nghĩa là xác thực nguồn gốc của thông điệp, phát hiện thông điệp có bị thay đổi, chỉnh sửa trong quá trình truyền hay không. Trong bài viết này chúng tôi sử dụng mật mã khối CBC-MAC để tính toán và xác thực MAC trên mỗi gói tin. CBC-MAC thì hiệu quả và nhanh chóng vì nó dựa trên một thuật toán mã hóa khối để giảm thiểu số lượng các mã hóa nguyên thủy phải thực hiện trong điều kiện bộ nhớ hạn chế trong mạng cảm biến không dây.

MAC được xem là một kiểm tra mã hóa an toàn của thông điệp. Bên gửi và bên nhận phải sử dụng cùng 1 khóa bí mật để tính toán MAC. Bên gửi tính toán giá trị MAC với đầu vào là khóa bí mật và payload, giá trị MAC được gắn trên mỗi gói tin khi truyền. Bên nhận tính toán lại giá trị MAC cũng với khóa bí mật và payload, so sánh 2 giá trị MAC nếu giống nhau thì gói tin được chấp nhận nếu không giống thì gói tin bị loại bỏ. Giá trị MAC rất khó tính ra nếu không có khóa bí mật. Nghĩa là nếu kẻ tấn công thay đổi thông điệp hoặc tiêm thêm thông điệp thì kẻ tấn công không thể tính toán ra được giá trị MAC tương ứng khi đó người nhận sẽ kiểm tra và loại bỏ những thông điệp này.

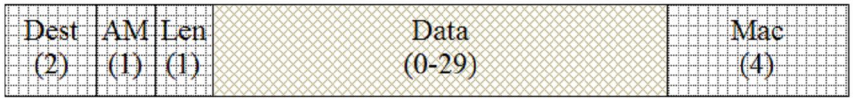
Sự an toàn của CBC-MAC liên quan trực tiếp đến độ dài của MAC. Giao thức bảo mật ở các máy thông thường sử dụng 8 hoặc 16 bytes MAC nhưng ở đây chúng tôi thấy có thể rút ngắn xuống còn 4 bytes MAC và chúng tôi nghĩ nó sẽ phù hợp trong bối cảnh các mạng cảm biến không dây vì: đầu ra của CBC-MAC là 4 bytes nhị phân, với 4 bytes chúng ta có 2^{32} giá trị khác nhau. Nếu kẻ tấn công tấn công bằng cách thử sai và thử lại thì kẻ tấn công có thể phải thử đến 2^{32} lần (tỷ lệ thành công là $1/2^{32}$) và để làm được điều này thì kẻ tấn công không thể thử offline mà chúng phải thử trực tiếp trên đường truyền để biết giá trị MAC có đúng hay không. Như vậy với kênh truyền 19,2 kbps của mạng cảm biến không dây mà phải gửi 2^{32} gói tin thì phải mất thời gian khoảng 25 tháng do đó đối với mạng cảm biến không dây khoảng thời gian như vậy có thể xem là an toàn (vì thời gian sống của nút cảm biến mica2 chỉ có khoảng vài tuần đến vài tháng).

2.2. Thực nghiệm

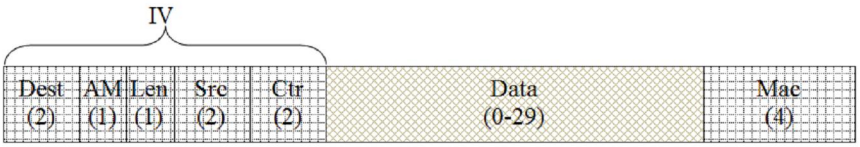
Các nút cảm biến được sử dụng phổ biến hiện nay như Mica, Mica2, Mica2Dot chạy trên nền hệ điều hành TinyOS. Với những kết quả như đã phân tích bài báo đã thực hiện việc bảo mật dữ liệu trên lớp liên kết dữ liệu bằng thuật toán mã hóa khối Skipjack và mã xác thực CBC-MAC để mã hóa và xác thực dữ liệu nhằm đảm bảo tính bảo mật, tính toàn vẹn và tính xác thực dữ liệu của các nút cảm biến trên hệ điều hành TinyOS.

Dựa trên định dạng gói tin hiện tại của TinyOS, tác giả đề xuất cải tiến về cấu trúc định dạng gói tin để phù hợp hơn với thuật toán CBC-MAC và thuật toán mã hóa khối như sau:

Hình 5. Định dạng gói tin mặc định trên TinyOS [5]



Hình 6. Định dạng gói tin với chế độ chỉ có xác thực



Hình 7. Định dạng gói tin với chế độ mã hóa và xác thực

Các gói tin ở lớp liên kết dữ liệu được mã hóa dữ liệu bằng thuật toán mã hóa khối Skipjack với vector khởi tạo (*IV*) có chiều dài là 8 bytes và xác thực bằng thuật toán CBC-MAC có chiều dài là 4 bytes MAC. Trong 8 bytes *IV* thực tế thì chỉ có thêm 2 bytes counter, các byte còn lại là được mượn từ phần header của gói tin gồm: Dest (2 bytes), AM (1 bytes), Len (1 byte), Src (2 bytes).

Bài báo đã thực hiện mã hóa và xác thực bằng 2 module riêng trên hệ điều hành TinyOS, các module được viết bằng ngôn ngữ nesC để tương thích với hệ điều hành TinyOS. Do 2 module mã hóa và xác thực được tách riêng nên đối với các ứng dụng chúng ta có thể lựa chọn một trong hai cách đó là: chỉ có xác thực mà không có mã hóa nghĩa là gói tin được xác thực bằng giá trị MAC nhưng phần dữ liệu payload của gói tin thì không được mã hóa; mã hóa và xác thực nghĩa là phần dữ liệu payload của gói tin được mã hóa bằng thuật toán Skipjack trước, sau đó mới tính giá trị MAC.

2.2.1. Mô tả thực nghiệm

Trên bộ mô phỏng TOSSIM có các thư viện hỗ trợ mô phỏng các nút cảm biến như Mica, Mica2, Mica2Dot, Mica128 chạy trên hệ điều hành TinyOS.

Trong thư viện TinyOS chúng tôi đã thêm 2 module mã hóa và xác thực bằng thuật toán Skipjack và CBC-MAC, trên ứng dụng chúng tôi thực hiện trên cảm biến Mica2 để

gửi dữ liệu. Ở đây chúng tôi gửi 24 bytes payload để đo năng lượng tiêu thụ ở mỗi chế độ: chế độ bảo mật của TinyOS hiện tại, chế độ chỉ có xác thực, chế độ mã hóa và xác thực. Đối với chế độ bảo mật TinyOS hiện tại dữ liệu được xác thực bằng thuật toán CRC, với chế độ xác thực thì dữ liệu được xác thực bằng thuật toán CBC-MAC được cung cấp bởi mudule xác thực, với chế độ mã hóa và xác thực thì dữ liệu được mã hóa trước bằng thuật toán Skipjack sau đó mới tính toán giá trị MAC để xác thực.

Hiện tại ở bài báo này thực nghiệm trên hệ điều hành TinyOS 1.1.13 có hỗ trợ để lấy thông số năng lượng tiêu thụ của mỗi nút cảm biến. Sau đó sẽ thu được các thông số năng lượng tiêu thụ của mỗi chế độ để phân tích.

2.2.2. Phân tích đánh giá kết quả đạt được

Với những phân tích ở trên bài báo đã đảm bảo về tính bảo mật, tính xác thực và tính toàn vẹn của các gói tin trên lớp liên kết dữ liệu. Tuy nhiên, theo hình 5, 6, 7 thấy rằng chiều dài của gói tin đề xuất so với chiều dài gói tin hiện hành của TinyOS là: tăng 1 byte đối với các gói tin chỉ có xác thực, tăng 5 bytes đối với các gói tin có xác thực và mã hóa. Sự tăng kích thước gói tin làm tăng sự tiêu hao năng lượng của các nút cảm biến vì phải tiêu tốn năng lượng cho sự tính toán và tốn thêm năng lượng cho việc gửi các gói tin có kích thước dài hơn. Điều này làm cho chúng ta phải cân nhắc khi sử dụng các ứng dụng mạng cảm biến không dây là nên lựa chọn bảo mật ở mức nào là thích hợp. Nếu chúng ta chọn ứng dụng có bảo mật là mã hóa và xác thực thì sẽ tiêu tốn năng lượng của các nút cảm biến nhiều dẫn đến thời gian sống của chúng giảm, ngược lại nếu chúng ta chỉ chọn là chỉ có xác thực không thì tiêu tốn năng lượng của các nút cảm biến sẽ ít hơn và thời gian sống của chúng tăng lên. Do đó, đòi hỏi chúng ta phải cân nhắc đánh đổi giữa sự bảo mật và mức tiêu hao năng lượng của các nút cảm biến.

Để đánh giá năng lượng tiêu hao, chúng tôi đã so sánh năng lượng tiêu hao khi sử dụng với chế độ bảo mật mặc định của TinyOS, bảo mật ở mức chỉ có xác thực, bảo mật ở mức mã hóa và xác thực. Bài viết đã thực hiện bằng cách gửi gói tin 24 bytes payload với TinyOS mặc định, bảo mật ở mức chỉ có xác thực, bảo mật ở mức mã hóa và xác thực thu được bảng so sánh như bảng 1.

Bảng 1. So sánh tổng năng lượng tiêu hao để gửi gói tin 24 byte payload với chế độ bảo mật của TinyOS

Chế độ bảo mật	Năng lượng tiêu hao (μAH)	Tỷ lệ % tiêu hao so với TinyOS
TinyOS hiện tại	1058250	0,00%
Chỉ có xác thực	1060650	0,23%
Mã hóa và xác thực	1070250	1,13%

Bảng 1 cho thấy rằng chế độ bảo mật chỉ có xác thực tăng 0,23% so với TinyOS hiện hành, chế độ mã hóa và xác thực tăng 1,13% so với TinyOS hiện hành. Sự tăng năng lượng này là do kích thước gói tin tăng phải tiêu tốn năng lượng để gửi và tiêu tốn năng lượng để tính toán.

3. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Bài viết trình bày các thuật toán mã hóa, xác thực và đã thực hiện trên mạng cảm biến không dây với bộ mô phỏng TOSSIM để bảo mật dữ liệu trong mạng cảm biến không dây nhằm đạt được tính bảo mật, tính xác thực và tính toàn vẹn dữ liệu trên lớp liên kết dữ liệu. Chúng tôi đã phân tích đánh giá mức độ bảo mật của các thuật toán sử dụng để đảm bảo bảo mật dữ liệu trong mạng cảm biến không dây, đưa ra kết quả thực nghiệm về mức tiêu hao năng lượng đối với các chế độ bảo mật để người dùng khi sử dụng có thể cân nhắc đánh đổi giữa các mức độ bảo mật và tiêu hao năng lượng (làm ảnh hưởng đến thời gian sống của các cảm biến). Hướng phát triển sẽ kiểm nghiệm so sánh về thời gian, băng thông giữa các chế độ bảo mật.

TÀI LIỆU THAM KHẢO

[1] Al-Sakib Khan Pathan, Hyung-Woo Lee, & Choong Seon Hong (2006), *Security in Wireless Sensor Networks: Issues and Challengesm,* International Conference on Advanced Communication Technology, 2(8),1043-1048.

[2] Azzedine Boukerche (2008), *Algorithms and Protocols for Wireless Sensor Networks*, Hoboken, New Jersey. John Wiley & Sons.

[3] Chris Kowalczyk. Block ciphers modes of operation, *Crypto-it*. Truy cập từ: www.crypto-it.net [19/09/2018].

[4] David Boyle, Thomas Newe (2009), *Securing Wireless Sensor Networks: Security Architectures,* Journal of networks, 3(1), 65-77.

[5] Javier Lopez, Jianying Zhou (2008), *Wireless Sensor Network Security*, Cryptology and Information Security Series.1.

[6] Lars Knudsen, David Wagner (2001), *On the structure of Skipjack,* Discrete Applied Mathematics, 111, 103-116.

[7] Parli B. Hari, Monika (2014), *Protocols Security for Wireless Sensor Networks,* International Journal of Science and Research, 3(10), 1231-1234.

[8] Ramnath Venugopalan, Prasanth Ganesan, Pushkin Peddabachagari, Alexander Dean, Frank Mueller, Mihail Sichitiu (2003), *Encryption overhead in embedded systems and sensor network nodes: Modeling and analysis,* International conference on Compilers, architecture and synthesis for embedded systems, 188-197.

[9] Saurabh Singh, Dr. Harsh Kumar Verma, (2011), *Security For Wireless Sensor Network*, International Journal on Computer Science and Engineering, 3(6), 2393-2399.

[10] Skipjack. Skipjack (cipher), *Wikipedia*. Truy cập từ: https://en.wikipedia.org [19/09/2018].

[11] *Tutorialspoint. Message Authentication. Tutorialspoint.com.* Truy cập từ http://www.tutorialspoint.com [19/09/2018].

SECURE COMMUNICATION IN WIRELESS SENSOR NETWORK

Le Hoang Anh

ABSTRACT

Wireless sensor network provides great benefits of low cost and flexibility, but it also poses many challenges to be addressed. Some of the main problems can be identified as follows: limited energy resources, restricted memory storage and restricted power of processor, especially the security mechanisms for wireless sensor network. Due to these limitations, security techniques in the traditional wireless network cannot be applied effectively on wireless sensor network. Therefore, the security issue in the wireless sensor network is considered as a major challenge that draws much intention from researchers. In this paper, we first discuss several advantages and challenges of wireless sensor network. We then propose two security measures: (1) authentication, (2) encryption and authentication. We also give an implementation to show that energy consumption is acceptable (with an overhead of 0.23%) and data is secured.

Keywords: *Safety data, sensor, security, wireless sensor network.*

Ngày nộp bài: 23/10/2018; Ngày gửi phản biện: 9/11/2018; Ngày duyệt đăng: 6/8/2019