Transport and Communications Science Journal

APPLICATION OF PROVERIF TO VERIFY THE SECURITY OF BLOCKCHAIN-BASED AUTHENTICATION AND KEY AGREEMENT PROTOCOLS FOR 5G NETWORKS

Tran Thi Nga*, Nguyen Duc Cong, Mai Duc Tho

Academy of Cryptogrhaphy Techniques, No 141 Chien Thang Street, Tan Trieu, Thanh Tri, Hanoi, Vietnam

ARTICLE INFO

TYPE: Research Article
Received: 03/03/2025
Revised: 21/04/2025
Accepted: 10/06/2025
Published online: 15/06/2025
https://doi.org/10.47869/tcsj.76.5.3

* *Corresponding author*
Email: trannguyetnga512@gmail.com; tranthinga@actvn.edu.vn

Abstract. The Authentication and Key Agreement (AKA) protocol plays a crucial role in ensuring security within 5G mobile network systems. However, the 5G-AKA protocol still faces several issues related to maintaining confidentiality, authentication, and integrity among users, service networks, and home networks. Many studies have been proposed to address the existing weaknesses of the 5G-AKA protocol. Blockchain technology, with its decentralized nature and strong resistance to tampering, has been applied to design more secure authentication protocols. This paper aims to verify the security of a blockchain-based 5G-AKA authentication and key agreement protocol by using the proVerif tool to analyze confidentiality, authentication, and equivalence properties. The analysis results show that the protocol satisfies critical security requirements, including session key confidentiality, mutual entity authentication, and resistance to common attacks such as man-in-the-middle and replay attacks. The study demonstrates the potential of blockchain technology in enhancing the security of 5G networks and emphasizes the essential role of formal verification tools like ProVerif in developing secure systems.

Keywords: Authentication and key agreement 5G-AKA, Blockchain, Proverif

Tạp chí Khoa học Giao thông vận tải

ỨNG DỤNG PROVERIF KIỂM TRA SỰ AN TOÀN CỦA GIAO THỨC XÁC THỰC VÀ THỎA THUẬN KHÓA DỰA VÀO BLOCKCHAIN CHO MẠNG 5G

Trần Thị Ngà*, Nguyễn Đức Công, Mai Đức Thọ

Học viện Kỹ thuật mật mã, Số 141 Chiến Thắng, Tân Triều, Thanh Trì, Hà Nội, Việt Nam

THÔNG TIN BÀI BÁO

CHUYÊN MỤC: Công trình khoa học
Ngày nhận bài: 03/03/2025
Ngày nhận bài sửa: 21/04/2025
Ngày chấp nhận đăng: 10/06/2025
Ngày xuất bản Online: 15/06/2025
https://doi.org/10.47869/tcsj.76.5.3

* *Tác giả liên hệ*
Email: trannguyetnga512@gmail.com; tranthinga@actvn.edu.vn

Tóm tắt. Giao thức xác thực và thỏa thuận khóa AKA có vai trò quan trọng trong việc đảm bảo an toàn trong hệ thống của mạng di động 5G. Tuy nhiên, giao thức 5G-AKA vẫn gặp phải nhiều vấn đề liên quan đến việc đảm bảo tính bí mật, xác thực và toàn vẹn giữa người dùng, mạng dịch vụ và mạng thường trú. Nhiều nghiên cứu đã đưa ra để khắc phục những yếu điểm còn tồn tại trong giao thức 5G-AKA. Công nghệ blockchain với tính chất phi tập trung và khả năng chống giả mạo, đã được ứng dụng để thiết kế các giao thức xác thực an toàn hơn. Bài báo này với mục đích kiểm tra an toàn giao thức xác thực và thỏa thuận khóa 5G-AKA dựa vào blockchain, bằng việc sử dụng công cụ proverif để phân tích các thuộc tính bí mật, xác thực và các thuộc tính tương đương. Kết quả phân tích cho thấy giao thức đáp ứng các yêu cầu bảo mật như đảm bảo bí mật khóa phiên, xác thực thực thể hai chiều và khả năng chống lại nhiều loại tấn công phổ biến như tấn công xen giữa và tấn công phát lại. Nghiên cứu cho thấy tiềm năng của blockchain trong việc nâng cao an ninh cho mạng 5G, đồng thời khẳng định vai trò thiết yếu của các công cụ phân tích hình thức như ProVerif trong việc phát triển các hệ thống an toàn.

Từ khóa: Giao thức xác thực và thỏa thuận khóa 5G-AKA, Blockchain, Proverif.

1. ĐẶT VẤN ĐỀ

Kiểm tra giao thức an toàn là một lĩnh vực nghiên cứu được nhiều nhà nghiên cứu quan tâm, bởi giao thức có ở khắp nơi và chúng được ứng dụng nhiều trong thương mại điện tử, mạng không dây, bỏ phiếu điện tử… Hơn nữa, các phương pháp kiểm tra chính thức cung cấp cách phân tích tính bảo mật của một giao thức trước và ngay cả khi nó được triển khai [1]. Điều này được minh chứng bằng các cuộc tấn công đã tìm thấy chống lại nhiều giao thức đã được công

bố như giao thức SSL/TLS được sử dụng cho kết nối https://. Phiên bản đầu tiên có từ năm 1994 và kể từ đó có nhiều cuộc tấn công được phát hiện [2]. Hơn nữa, các lỗi bảo mật không được phát hiện bằng thử nghiệm chức năng, vì chúng chỉ xuất hiện khi có sự hiện diện của kẻ tấn công, những lỗi này có thể gây ra hậu quả nghiệm trọng đối với an toàn của hệ thống. Do đó, việc kiểm tra giao thức có an toàn hay không trước khi đưa vào sử dụng sẽ góp phần đáng kể giúp giảm thiểu các tấn công có thể xảy ra trong thực tế. Từ đó, giúp các nhà thiết kế có thể cải thiện và nâng cao an toàn cho giao thức.

Hai mô hình được sử dụng để kiểm tra giao thức là mô hình tượng trưng (Symbolic Model) và mô hình tính toán (Computationa Model). Trong mô hình tính toán các nguyên thủy mật mã là các hàm từ các chuỗi tới các chuỗi và kẻ tấn công là một loại máy turing có thể đưa ra các quyết định dựa trên xác suất [2]. Mô hình tượng trưng là một mô hình trừu tượng sử dụng các công cụ để kiểm tra tự động, nhiều công cụ được sử dụng đã tồn tại như: AVISPA [3], FDR [4], Scyther [5], Tamarin [6], Proverif [2], [7], [8], [9]... Các công cụ kiểm tra giao thức cung cấp cách thức để mô hình hóa và kiểm tra các thuộc tính của giao thức mà không cần chứng minh thủ công. Quá trình này thường bao gồm việc mô hình hóa giao thức với một số ngôn ngữ đầu vào và chỉ định các thuộc tính an toàn mong muốn, công cụ sau đó cố gắng chứng minh hoặc bác bỏ thông qua tìm kiếm không gian trạng thái hoặc các phương pháp khác nhau. Khi một thuộc tính được chứng minh là sai nghĩa là tìm thấy một dẫn xuất thực thi mâu thuẫn, truy vết sẽ được trả về như một ví dụ để bác bỏ thuộc tính [1]. Trong bài báo này, nhóm tác giả sử dụng công cụ proverif để kiểm tra an toàn giao thức vì proverif là trình kiểm tra giao thức tượng trưng tự động. Nó hỗ trợ một loạt các nguyên thủy mật mã, được xác định bằng quy tắc viết lại hoặc bằng các phương trình. Nó có thể chứng minh các thuộc tính an toàn khác nhau: tính bí mật, tính xác thực và thuộc tính tương đương (ví dụ: tính riêng tư), với không gian thông báo không giới hạn và số lượng phiên không giới hạn.

Mạng 5G hiện nay đang được thế giới sử dụng rộng rãi để truy cập các dịch vụ mạng di động và kết nối vạn vận IoT (Internet of things). Để bảo mật thông tin liên lạc 5G, dự án đối tác thế hệ thứ ba 3GPP (3rd Generation Partnership Project) đã chuẩn hóa giao thức 5G-AKA (5G Authentication and Key Agreement*)* là phiên bản cải tiến của giao thức AKA hiện đang được sử dụng trong 4G EPS-AKA (Evolved Packet System) [10]. 5G-AKA có thể xác thực lẫn nhau giữa thiết bị người dùng UE (User Equipment), mạng dịch vụ SN (Serving Network), và mạng thường trú (Home Networks). Nó đã giải quyết một số vấn đề bảo mật đã tồn tại từ trước được tìm thấy trong các mạng 4G. Tuy nhiên, giao thức 5G-AKA hiện tại phải chịu nhiều mối đe dọa bảo mật ảnh hưởng đến độ tin cậy của mạng 5G. Do đó, nhiều công trình nghiên cứu đã đưa ra để khắc phục những điểm yếu của giao thức 5G-AKA [11]. Gần đây giải pháp sử dụng công nghệ blockchain đã được sử dụng để khắc phục các yếu điểm đã chỉ ra trong mạng 5G-AKA. Yang cùng cộng sự [12] đã giới thiệu ý tưởng về một lược đồ truy cập ẩn danh dựa trên blockchain (Blockchain-based Anonymous Access) cho phép các nhà sản xuất thiết bị, nhà điều hành mạng và người dùng truy cập vào cơ sở dữ liệu dựa trên blockchain và thực hiện xác thực lẫn nhau. Họ đề xuất rằng tất cả các điểm truy cập AP (Access Point) 5G có thể sử dụng khóa công khai của UE được liệt kê trong blockchain để thực hiện xác thực lẫn nhau giữa AP và UE. Tuy nhiên, điều này thiếu một cơ chế để UE truy xuất khóa công khai của các AP xung quanh. Xu cùng cộng sự đã đề xuất sử dụng blockchain có thể chỉnh sửa lưu trữ tất cả thông tin của người đăng ký [13]. Blockchain có thể chỉnh sửa cung cấp các chức năng xóa và thu hồi khóa. Điều này có lợi cho các nhà điều hành mạng trong việc bảo vệ quyền riêng tư của người dùng. Tuy nhiên, đề xuất thiếu giao thức xác thực cho UE và mạng lõi để bảo mật dữ liệu người dùng bằng cách sử dụng khóa trong Blockchain. Trong tài liệu [14] các tác giả giới thiệu về việc sử dụng blockchain giúp cho các dịch vụ chuyển vùng trong mạng 5G. Tuy nhiên, bài báo lại

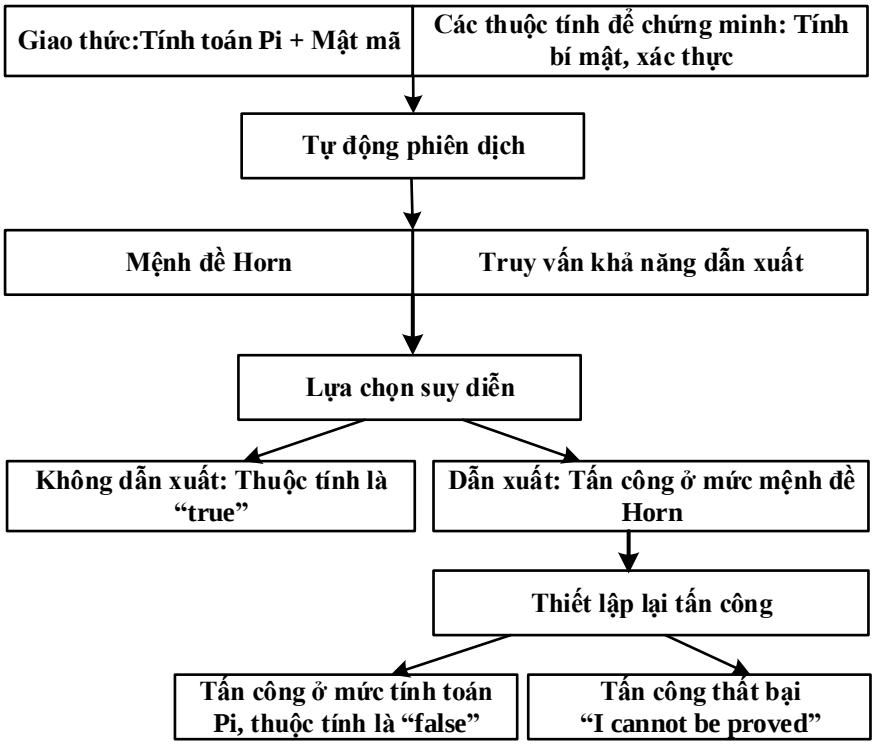
không phân tích làm rõ kết quả đầu ra khi có dấu hiệu tấn công mà lại lựa chọn bỏ qua.

Qua khảo sát trên có thể thấy việc nghiên cứu giao thức 5G-AKA vẫn còn tồn tại những lỗ hổng trong bản thân giao thức. Do đó, việc sử dụng phương pháp tượng trưng để kiểm tra giao thức có an toàn hay không trước khi triển khai trong thực tế có ý nghĩa rất quan trọng góp phần vào thành công của hệ thống. Đặc biệt, với Việt Nam khi mạng 5G đang được các nhà mạng như Viettel,VNPT và MobiFone bắt đầu triển khai, thì việc nghiên cứu các vấn đề liên quan đến an ninh, an toàn cho mạng 5G sẽ góp phần thúc đẩy quá trình chuyển đổi số của đất nước. Với mong muốn, hiểu hơn về công cụ proverif và cách thức hoạt động của giao thức 5G-AKA dựa vào blockchain. Nhóm tác giả lựa chọn giao thức 5G-AKA dựa vào blockchain ứng dụng proverif để kiểm tra các thuộc tính: tính bí mật, tính toàn vẹn và các thuộc tính tương đương nhằm đảm bảo an toàn cho mạng 5G.

2. PHƯƠNG PHÁP NGHIÊN CỨU

2.1. Cấu trúc của Proverif

Cấu trúc của ProVerif được trình bày trong Hình 1, ProVerif nhận đầu vào là mô tả của giao thức bằng cách áp dụng cú pháp tính toán *pi* nhằm kiểm tra các thuộc tính bảo mật [1], [2], [7], [8]. Các thuộc tính bảo mật được mô hình hóa như các truy vấn về khả năng dẫn xuất. Sau đó, đầu vào được tự động dịch thành một tập hợp các mệnh đề *horn* [8], [9]. Mệnh đề *horn* là quy tắc lập trình logic, được kết hợp bởi phép logic hoặc phép toán và được định nghĩa là: $F_1 \wedge ... \wedge F_n \Rightarrow F$; F được gọi là một sự kiện, định nghĩa này nghĩa là nếu tất cả các sự kiện F_i (với i=1,…, n) là đúng, thì F cũng đúng.



Hình 1. Cấu trúc của Proverif.

Các mệnh đề này phải tuân theo thuật toán phân giải với các sự lựa chọn để xác định liệu một sự kiện có được dẫn xuất từ các mệnh đề hay không. Nếu sự kiện không thể được dẫn xuất

từ mệnh đề thì thuộc tính bảo mật mong muốn được chứng minh. Nếu sự kiện được dẫn xuất từ các mệnh đề, thì có thể có một cuộc tấn công chống lại thuộc tính đang xét: dẫn xuất này có thể tương ứng với một cuộc tấn công nhưng cũng có thể tương ứng với một cuộc tấn giả. Thuật toán phân giải sử dụng một số tối ưu như: loại bỏ mệnh đề phụ, loại bỏ các giả thuyết trùng lặp, loại bỏ mệnh đề mà kết luận đã có trong giả thuyết… nhằm tăng tốc cho thuật toán phân giải, những tối ưu này được áp dụng trên các mệnh đề ban đầu và sau mỗi bước giải quyết [2], [8]. Proverif thực thi mô hình tấn công Dolev-Yao tiêu chuẩn, nghĩa là tất cả thông tin liên lạc đều đi qua kẻ tấn công trừ khi một kênh được khai báo là *private*.

Proverif chứng minh hoặc bác bỏ các thuộc tính của giao thức có thể thực hiện hoàn toàn tự động hoặc sử dụng chế độ tương tác. Ở chế độ tự động, proverif sử dụng chiến lược giải quyết mặc định để chứng minh các thuộc tính cung cấp mà không cần sự tương tác của người dùng. Ngược lại, chế độ tương tác yêu cầu người dùng chủ động hỗ trợ quá trình kiểm tra bằng cách hướng dẫn công cụ theo cách thủ công.

Công cụ có thể được sử dụng để chứng minh các thuộc tính khả năng tiếp cận, khẳng định tương ứng và tương đương quan sát. Thuộc tính dấu vết thể hiện khả năng tiếp cận các trạng thái giao thức. Chúng thường được sử dụng để kiểm tra tính đúng đắn của mô hình và phân tích hiểu biết của kẻ tấn công. Mặt khác, các khẳng định tương ứng đề cập tới các mối liên quan và trật tự của các sự kiện. Ví dụ, chúng có thể sử dụng để kiểm tra các thuộc tính xác thực bằng cách phân tích chuỗi sự kiện trong dấu vết giao thức. Cuối cùng, tương đương quan sát đề cập tới hai quá trình không thể phân biệt được với quan điểm của kẻ tấn công, ví dụ: thể hiện quyền riêng tư.

2.2. Giao thức 5G-AKA dựa vào blockchain

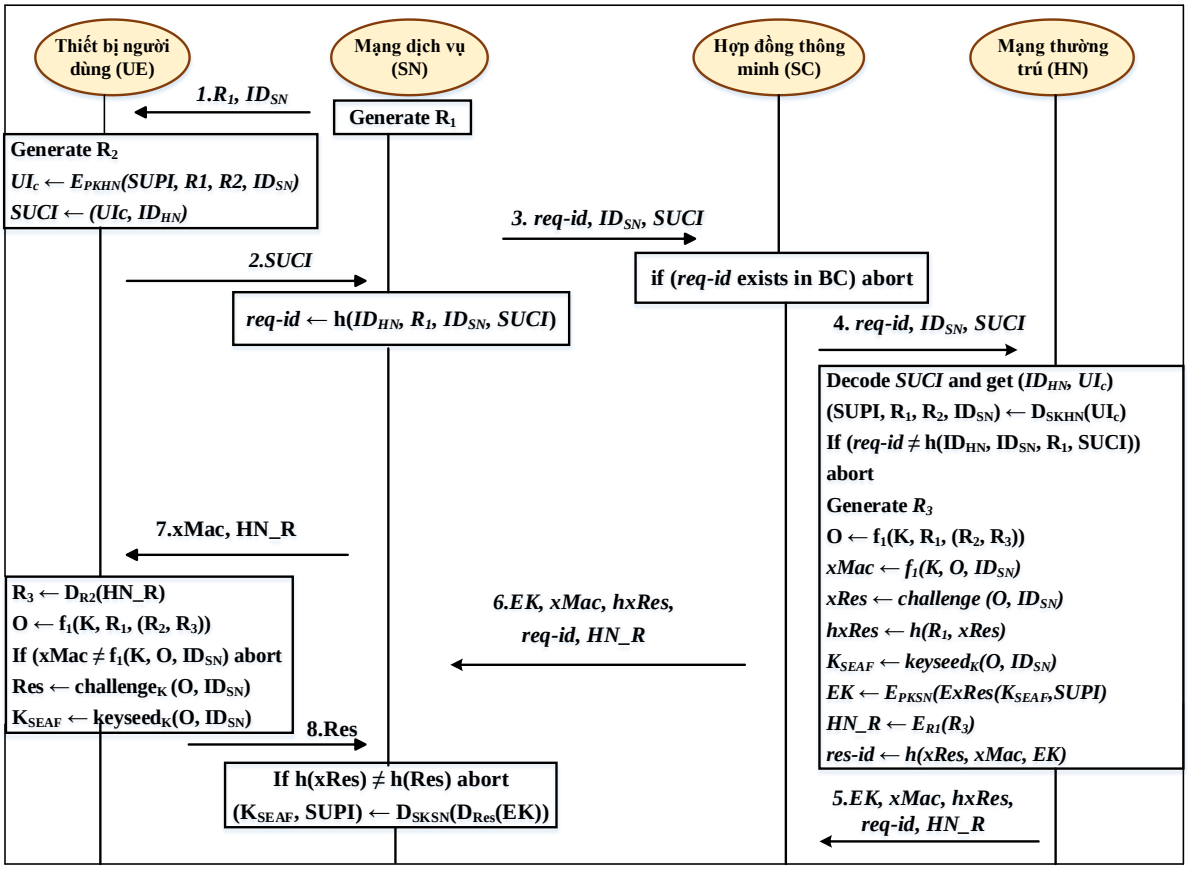
Trong phần này bài báo sử dụng mô hình trong [14] để kiểm tra giao thức 5G-AKA dựa vào blockchain có an toàn hay không.

Bảng 1. Tổng hợp các định nghĩa và ký hiệu được sử dụng trong giao thức.

Ký hiệu	**Mô tả**
UE	Thiết bị người dùng
SN	Mạng dịch vụ
HN	Mạng thường trú
SC	Hợp đồng thông minh
ID_U	Định danh của người dùng
R_i	Số ngẫu nhiên
Res	Phản hồi thách thức của người dùng
hxRes	Hàm băm phản hồi được mong đợi
xMac	Mã xác thực thông báo mong đợi
req_id	Định danh duy nhất cho yêu cầu xác thực
res_id	Định danh duy nhất cho phản hồi xác thực
PK_U	Khóa công khai của bên U
SK_U	Khóa riêng của bên U
SUPI	Định danh dài hạn của thuê bao
SUCI	Định danh ẩn của thuê bao
h(.)	Hàm băm
$f_1(K,…)$	Khóa nguyên thủy hàm một chiều
$Keyseed_K(…)$	Khóa nguyên thủy hàm một chiều sử dụng để sinh khóa phiên

$challenge_K$	Khóa nguyên thủy hàm một chiều để sinh đầu ra
$E_K(m)$	Mã hóa bất đối xứng thông điệp m với khóa công khai K
$D_K(c)$	Giải mã bất đối xứng bản mã c với khóa K
$\varepsilon_K(m)$	Mã hóa đối xứng m với khóa K
$D_K(c)$	Giải mã đối xứng c với khóa K

Giao thức 5G-AKA dựa vào blockchain được mô tả chi tiết trong hình 2 gồm 4 thực thể: UE (User Equipment), SN (Home Network), SC (Smart Contract) và HN (Home Network). Để bắt đầu giao thức mỗi HN sẽ tạo SC công khai của riêng mình và công bố địa chỉ của SC trên trang mạng của mình. Mỗi SN muốn cung cấp dịch vụ chuyển vùng cho UE đến HN sẽ phải liên lạc thông qua SC.



Hình 2. Giao thức 5G-AKA dựa vào blockchain.

Bước 1: SN tạo ra một số ngẫu nhiên (R_1) và gửi cho UE kèm theo định danh của mình ID_{SN}.

Bước 2: Khi nhận được yêu cầu, UE tạo ra một số ngẫu nhiên (R_2). UE sử dụng khóa công khai của HN là PK_{HN} để mã hóa SUPI, R_1, R_2 kết hợp với định danh ID_{SN} tạo thành UI_C. Bắt đầu tạo thông báo phản hồi SUCI bằng cách kết hợp định danh ID_{SN} và UI_C. Sau đó, UE gửi SUCI lại cho SN như công thức (1).

$$\begin{aligned} UI_C &\leftarrow \varepsilon_{PK_{HN}}(SUPI, R_1, R_2, ID_{SN}) \\ SUCI &\leftarrow (UI_C, ID_{HN}) \end{aligned} \tag{1}$$

Bước 3: Sau khi nhận được thông báo phản hồi của UE. SN tính toán giá trị băm của ID_{SN}, R_1, ID_{HN} và SUCI để được định danh duy nhất với yêu cầu xác thực ký hiệu là *req-id* theo công thức (2).

$$req-id \leftarrow h(ID_{HN}, R_1, ID_{SN}, SUCI) \tag{2}$$

Sau đó SN gửi yêu cầu xác thực bao gồm req-id, ID_{SN}, SUCI đến HN bằng cách đăng ký yêu cầu hành động giao dịch với SC của HN.

Bước 4: Khi nhận được yêu cầu, hợp đồng thông minh sẽ kiểm tra tính mới của yêu cầu, bằng cách tìm kiếm các bản ghi xác thực với req-id để tránh yêu cầu trùng lặp.

Nếu yêu cầu là yêu cầu mới, thông điệp sẽ được chuyển đến HN, ngược lại yêu sẽ bị từ chối. Chức năng SC có nhiệm vụ ngăn chặn các cuộc tấn công phát lại có nguồn gốc từ SN độc hại.

Bước 5: Khi nhận được yêu cầu từ SC, HN thực hiện kiểm tra định danh của mình với ID_{HN} chứa trong SUCI. HN giải mã UI_C bằng khóa riêng của mình nhận được SUPI, R_1, R_2, ID_{SN}. Đồng thời, so sánh các giá trị vừa nhận được với req-id. Quá trình xác thực sẽ dừng lại nếu bất kỳ điều kiện nào không thỏa mãn. Ngược lại, HN sẽ tiếp tục hành động với các hành động sau:

HN tạo số ngẫu nhiên R_3, tiếp tục lấy đầu vào là R_1, R_2, R_3 vào hàm f_1, tính toán giá trị xMac của thông điệp bằng cách đưa *O* và ID_{SN} vào hàm f_1 với khóa K như công thức (4).

$$\begin{aligned} O &\leftarrow f_1(K, R_1, R_2, R_3) \\ xMac &\leftarrow f_1(K, ID_{SN}, O) \end{aligned} \tag{4}$$

Tính toán xRes bằng cách sử dụng hàm challenge với khóa K và đưa *O*, ID_{SN} làm đầu vào. Tạo giá trị hxRes được tạo dưới dạng hàm băm của R_1 và xRes. Tính toán K_{SEAF} bằng cách đưa *O* và ID_{SN} vào hàm keyseed theo công thức (5).

$$\begin{aligned} xRes &\leftarrow challenge_K(O, ID_{SN}) \\ hxRes &\leftarrow h(R_1, xRes) \\ K_{SEAF} &\leftarrow keyseed_K(O, ID_{SN}) \end{aligned} \tag{5}$$

Tạo EK với mã hóa kép. Đầu tiên, mã hóa đối xứng khóa phiên và SUPI bằng xRes. Sau đó, mã hóa bất đối xứng bằng khóa công khai của SN như công thức (6).

$$EK \leftarrow E_{PK_{SN}}(\varepsilon_{xRes}(K_{SEAF}, SUPI)) \tag{6}$$

Tạo HN_R bằng cách mã hóa R_3 với mã hóa đối xứng với khóa là R_2.

Tạo định danh duy nhất cho thông báo phản hồi res-id bằng cách băm giá trị hxRes, xMac, EK theo công thức (7).

$$res-id \leftarrow h(hxRes, xMac, EK) \tag{7}$$

Cuối cùng, HN đăng ký giao dịch trên Blockchain chứa EK, xMac, hxRes, req-id, res-id và HN_R bằng chức năng hợp đồng thông minh tương ứng.

Ghi chú: f_1, challenge và key seed là các hàm mã hóa nguyên thủy được định nghĩa trên HN và chúng được nhúng trong thẻ USIM. Vì EK thu được thông qua việc mã hóa thông tin người dùng bằng khóa xRes, thông tin liên quan đến người dùng chỉ có thể truy cập được bởi

SN sau khi xác thực thành công. R_3 được mã hóa bằng cách sử dụng khóa R_2 vì đó là cách duy nhất mà một UE có thể dễ dàng thu được R_3.

Bước 6: SC kiểm tra xem người gửi thông điệp có phải là chủ sở hữu của SC hay không. Vì chỉ HN mới được phép đăng ký giao dịch phản hồi với SC. Nếu đúng, SC gửi toàn bộ thông điệp sang SN. Ngược lại, giao thức sẽ dừng.

Bước 7: Sau khi SN nhận được phản hồi của HN thông qua SC tương ứng. SN chuyển tiếp các phần xMac và HN_R cho UE. SN lưu trữ các giá trị còn lại để sử dụng sau khi nhận được phản hồi từ UE.

Bước 8: Sau khi nhận được phản hồi từ SN. UE thực hiện giải mã HN_R và tính toán giá trị *O* và *Mac* theo công thức (8) :

Giải mã HN_R bằng khóa R_2 để nhận được R_3

Tính giá trị *O* bằng cách cung cấp R_1, R_2, R_3 làm đầu vào hàm f_1.

Tính toán giá trị Mac bằng cách cung cấp O, ID_{SN} làm đầu vào hàm f_1.

$$\begin{aligned} R_3 &\leftarrow D_{R_2}(HN\_R) \\ O &\leftarrow f_1(K, R_1, (R_2, R_3) \\ Mac &\leftarrow f_1(K, ID_{SN}, O) \end{aligned} \tag{8}$$

Sau đó, UE kiểm tra Mac vừa tính toán với xMac nhận được từ SN. Nếu khớp với nhau quá trình xác thực được chập nhận. UE tiếp tục tính Res và K_{SEAF} theo công thức (9):

Tính Res bằng cách sử dụng hàm challenge với đầu vào là *O* và ID_{SN}

Tính toán khóa K_{SEAF} bằng cách sử dụng chức năng keyseed với đầu vào là *O* và ID_{SN}.

$$\begin{aligned} Res &\leftarrow challenge_K(O, ID_{SN}) \\ K_{SEAF} &\leftarrow keyseed_K(O, ID_{SN}) \end{aligned} \tag{9}$$

Cuối cùng, UE gửi giá trị Res tới SN. Khi nhận được phản hồi từ UE. SN tính toán giá trị băm của Res và so sánh với hxRes. Nếu kiểm tra là đúng, SN xác thực được UE. Bây giờ, có thể giải mã EK, đầu tiên bằng việc giải mã đối xứng bằng Res sau đó giải mã bất đối xứng của kết quả bằng khóa riêng của mình để thu được K_{SEAF} và SUPI tương ứng. Từ lúc này, UE có thể giao tiếp với SN bằng khóa phiên đã thiết lập K_{SEAF}.

Ghi chú: Quá trình xác thực sẽ bị hủy bỏ nếu bất kỳ bước kiểm tra, so sánh nào không đúng.

2.3. Mô hình hóa giao thức 5G-AKA dựa vào blockchain

Mô hình gồm ba phần chính: Khai báo, xử lý macros và xử lý chính. Để bắt đầu giao thức, HN và SN sẽ sinh cặp khóa công khai và bí mật. Đầu tiên, cặp khóa được sử dụng để mã hóa và giải mã thông báo. Thứ hai, cặp khóa được sử dụng để ký thông điệp và kiểm tra chữ ký. Tuy nhiên, ở đây không xét đến sử dụng chữ ký điện tử trong thông điệp của nó, và điều này được thực hiện ngầm định giữa các bên tham gia khi giao dịch được đăng ký trong blockchain. Cụ thể hơn, nó được thực hiện khi SN đăng ký yêu cầu xác thực hoặc khi HN đăng ký phản hồi xác thực.

Khai báo

Phần khai báo gồm ba phần: phần dành cho các kiểu người dùng định nghĩa, phần dành

cho định nghĩa đối tượng và phần dành cho các hàm được định nghĩa.

**** Thuật toán 1: Định nghĩa loại, đối tượng và hàm chức năng***

```
type UEKey.
type Key.
type publicKey.
type privateKey.

free bcChannel: channel. (*Public*)
free airChannel: channel. (*Public*)
free SUPI: bitstring [private].
free K: UEKey [private].
free KSEAF: bitstring [private].
free R3: bitstring [private].
free R2: bitstring [private].

fun hash(bitstring): bitstring.
fun mKey (bitstring): Key.
fun pk (privateKey): publicKey.
fun pkiEnc (bitstring, publicKey): bitstring.
reduc forall m: bitstring, k: privateKey; pkiDec (pkiEnc (m, pk (k)), k) = m.
fun symEnc (bitstring, Key): bitstring.
reduc forall m: bitstring, k: Key; symDec (symEnc (m, k), k) = m.
fun sign (bitstring, privateKey): bitstring.
reduc forall m: bitstring, k: privateKey; checksign(sign(m, k), pk (k)) = m.
fun decode(bitstring): bitstring.
fun f1 (UEKey, bitstring, bitstring): bitstring.
fun challenge(UEKey, bitstring, bitstring): bitstring.
fun keyseed(UEKey, bitstring, bitstring): bitstring.
```

Phần đầu tiên, *privateKey* và *publicKey* khởi tạo cặp khóa riêng và khóa công khai sử dụng mã hóa bất đối xứng và chữ ký số. Trong khi *Key* và *UEKey* là những loại được sử dụng cho khóa mã đối xứng.

Trong phần thứ hai, hai câu lệnh đầu tiên khai báo một cặp kênh giao tiếp*: airChannel* và *bcChannel*, các kênh này được sử dụng để mô hình hóa tương ứng giữa UE với SN và giữa SN với HN.

Sự hiện diện của từ khóa *free* trong Proverif là tương tự như định nghĩa biến toàn cục trong các ngôn ngữ lập trình khác, hay *free* được biết đến toàn bộ trong quá trình xử lý. Định nghĩa đối tượng được kết thúc bằng phương pháp truy cập xác định bằng từ khóa *[public]* hoặc [*private*]. Tức là các đối tượng mà kẻ tấn công không được biết phải được khai báo là *private*, trong khi những cái mà kẻ tấn công được biết sẽ được khai báo là *public.* Theo mặc định phương pháp truy cập là công khai (ví dụ: khi nó không được khai báo rõ ràng, công cụ sẽ coi nó là công khai). Sự hiện diện của các câu lệnh *free* với phương thức truy cập công khai trong định nghĩa *airChannel* và *bcChannel* cho thấy rằng cả hai kênh kẻ tấn công đều biết. Do đó, kẻ tấn công có thể nghe trộm bất cứ thông tin trao đổi nào qua 2 kênh đó. Định nghĩa này phản ánh giả định rằng kênh liên lạc giữa HN và SN được cung cấp bởi blockchain công khai. Vì trong blockchain công khai mỗi nút mong muốn có thể tham gia vào mạng blockchain và hoạt động như một nút đầy đủ, do đó có quyền truy cập dữ liệu giao dịch như các SC.

Đối tượng *SUPI, K, K_{SEAF}, R_3, và R_2*, tương ứng biểu thị định danh duy nhất của USIM, khóa lưu trữ trong USIM, khóa phiên, số ngẫu nhiên HN và SN. Từ khóa *[private]* trong định nghĩa ngụ ý rằng khóa là riêng và do đó kẻ tấn công không thể truy cập được.

Phần thứ ba định nghĩa các ký hiệu hàm bao gồm hàm tạo và hàm hủy. Trong Proverif, mỗi hàm tạo xây dựng các điều khoản bắt buộc để mô hình hóa một nguyên thủy cụ thể của giao thức mật mã. Trong giao thức *hash, symEnc, symDec, pkiEnc, pkiDec, sign và getPK* là hàm tạo, tương ứng mô hình hóa các hàm băm mật mã, mã hóa đối xứng, giải mã đối xứng, mã hóa khóa công khai, chữ ký số, và trả về khóa công khai của khóa bí mật.

Xử lý marcros

Thay vì giao thức được mô tả trong một chương trình chính, chúng tôi sử dụng chương trình con để khai báo sự tương tác giữa các bên tham gia. Vì giao thức có ba bên tham gia như những người chơi tích cực, ngoài chương trình chính. Mỗi quá trình xác định hành động của người tham gia tương ứng với các sự kiện xảy ra. Lưu ý, thay vì mô hình hóa blockchain như một xử lý macro, thì proverif mô hình hóa bằng kênh *bcChannel.*

Thuận toán 2: Quá trình xử lý của SN thể hiện như hình 3

```
let SN(IDSN:bitstring, IDHN:bitstring, pkHN:publicKey, pkSN:publicKey, skSN:privateKey) =
   new R1:bitstring;
   event SNSendReqToUE(IDSN);  (* Gửi thông tin quảng bá danh tính của SN *)
   out(airChannel, (R1, IDSN));
   in(airChannel, SUCI:bitstring);
   let recv_idHN = decode(SUCI) in
   if recv_idHN = IDHN then event SNRecvUERes(SUCI);
   let req_id = hash((IDHN, R1, IDSN, SUCI)) in
   let SignSN = sign(req_id, skSN) in
   event SNSendReqToHN(req_id);
   out(bcChannel, (req_id, SignSN, SUCI));
   in(bcChannel, (EKC:bitstring, xMac:bitstring, hxRes:bitstring, SignHN:bitstring, res_id:bitstring,
HN_R:bitstring));
   let(=pkHN, veri_req_id:bitstring, veri_res_id:bitstring) = checksign(SignHN, pkHN) in
   if (veri_res_id = res_id) then if (veri_req_id = req_id) then
   event SNRecvHNRes(SignHN);
   event SNSendReq2ToUE(xMac);
   out(airChannel, (xMac, HN_R));
   in(airChannel, Res:bitstring);
   if (hxRes = Res) then event SNRecvUERes2(Res);
   let EK = pkiDec(EKC, skSN) in
   let session_info = symDec(EK, mKey(Res)) in
   let sn_KSEAF = decode(session_info) in
   let sn_SUPI = decode(session_info).
```

Hình 3. Quá trình xử lý tại SN.

Trong thuật toán 2 hình 3, SN khởi tạo giao thức bằng việc lựa chọn một số ngẫu nhiên R_1 và chuyển cặp (R_1, ID_{SN}) vào giao diện *airChannel*, chờ phản hồi người dùng. Khi nhận được từ UE, SN tính *req_id* sử dụng hàm băm và *SignSN* bằng chữ ký số. Sau đó, SN chuyển *req_id, SignSN* và *SUPI* trên giao diện *bcChannel* và đợi HN phản hồi.

Khi nhận được phản hồi của HN, SN chọn (*xMac, HN_R*) từ thông điệp và chuyển chúng trên giao diện *airChannel*, chờ phản hồi từ người dùng. Khi nhận được phản hồi *Res* từ người dùng, SN so sánh *Res* với *hxRes* nhận được từ HN. Trong trường hợp trùng khớp, xác thực được chấp nhận và tiếp theo SN có thể truy xuất K_{SEAF} và *SUPI* từ phản hồi của HN.

Tương tự, các quy trình UE và HN được giải thích tương ứng trong thuật toán 3 hình 4 và thuật toán 4 hình 5.

Thuật toán 3: Quá trình xử lý tại UE như thể hiện trong hình 4

```
let UE(IDHN:bitstring, pkHN:publicKey, K:UEKey, SUPI:bitstring) =
    in(airChannel, (R1:bitstring, IDSN:bitstring));
    event UERecvSNReq(IDSN);
    (* new R2:bitstring; *)
    let UI = pkiEnc((SUPI, R1, R2, IDSN), pkHN) in
    let SUCI = (UI, IDHN) in
    event UESendResToSN(SUCI);
    out(airChannel, SUCI);
    in(airChannel, (xMac:bitstring, HN_R:bitstring));
    let ue_R3 = symDec(HN_R, mKey(R2)) in
    let O = f1(K, R1, (R2, R3)) in
    let Mac = f1(K, O, IDSN) in
    if (Mac = xMac) then event UERecvSNReq2(xMac);
    let Res = challenge(K, O, IDSN) in
    let ue_KSEAF = keyseed(K, O, IDSN) in
    event UESendRes2ToSN(Res);
    out(airChannel, Res).
```

Hình 4. Quá trình xử lý tại UE.

Thuật toán 4: Quá trình xử lý tại HN thể hiện trong hình 5

```
let HN(IDHN:bitstring, IDSN:bitstring, pkHN:publicKey, skHN:privateKey, pkSN:publicKey, K:UEKey) =
    in(bcChannel, (req_id:bitstring, signSN:bitstring, SUCI:bitstring));
    let(=pkSN, veri_req_id:bitstring) = checksign(signSN, pkSN) in
    if (veri_req_id = req_id) then event HNRecvSNReq(req_id, signSN);
    let UI = decode(SUCI) in
    let dec_UI = pkiDec(UI, skHN) in
    let HN_R2 = decode(dec_UI) in
    let R1 = decode(dec_UI) in
    let HN_SUPI = decode(dec_UI) in
    (* new R3:bitstring; *)
    let O = f1(K, R1, (HN_R2, R3)) in
    let xMac = f1(K, O, IDSN) in
    let xRes = challenge(K, O, IDSN) in
    let hxRes = hash((R1, xRes)) in
    let HN_KSEAF = keyseed(K, O, IDSN) in
    let EK = symEnc((HN_KSEAF, HN_SUPI), mKey(xRes)) in
    let EKC = pkiEnc(EK, pkSN) in
    let HN_R = symEnc(R3, mKey(R2)) in
    let res_id = hash((hxRes, xMac, EKC)) in
    let SignHN = sign((req_id, res_id), skHN) in
    event HNSendResToSN(req_id, res_id);
    out(bcChannel, (EKC, xMac, hxRes, SignHN, req_id, res_id, HN_R)).
```

Hình 5. Quá trình xử lý tại HN.

Trong thuật toán 4 hình 5, quá trình xử lý chính bắt đầu bằng quy trình xử lý từ khóa, tạo ra các khóa bất đối xứng riêng tư *skSN* và *skHN* tương ứng cho các chủ thể SN và HN. Sau đó kết quả được thể hiện công khai trên giao diện *bcChannel* và *airChannel*, đảm bảo các khóa công khai có thể được truy cập bởi bất kỳ kẻ tấn công nào. Hơn nữa, nó tạo ra các số nhận dạng

ID_{HN} và ID_{SN} tương ứng cho các chủ thể SN và HN.

Thuật toán 5: Đặc tính bảo mật và quá trình xử lý chính

query attacker (*SUPI*).
query attacker (*KSEAF*).
query attacker (*R3*).
query attacker (*R2*).
query attacker (*K*).

query x:bitstring; **event**(UERecvSNReq(x)) ==> **event**(SNSendReqToUE).
query x:bitstring, y:bitstring; **event**(SNRecvUERes(x)) ==> **event**(UESendResToSN(y)).
query x:bitstring, y:bitstring, z:bitstring;
event(HNRecvSNReq(x,y)) ==> **event**(SNSendReqToHN(z)).
query x:bitstring, y:bitstring, z:bitstring;
event(SNRecvHNRes(x)) ==> **event**(HNSendResToSN(y,z)).
query x:bitstring, y:bitstring; **event**(UERecvSNReq2(x)) ==> **event**(SNSendReq2ToUE(y)).
query x:bitstring; **event**(SNRecvUERes2) ==> **event**(UESendRes2ToSN(x)).

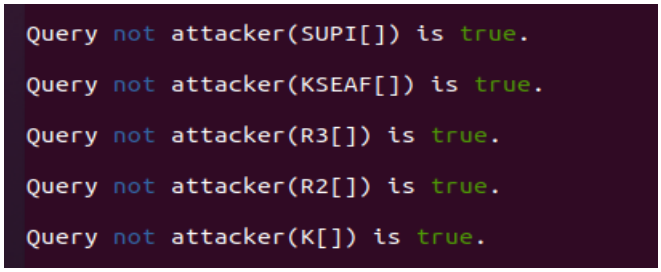
Process
new $skHN$: privateKey;
new $skSN$:privateKey;
new $IDHN$:bitstring;
new $IDSN$:bitstring;
let $pkHN$ = pk($skHN$) in **out** (airChannel, $pkHN$);
let $pkSN$ = pk($skSN$) in **out** (bcChannel, $pkSN$);
(!SN($IDSN$, $IDHN$, $pkHN$, $pkSN$, $skSN$) |
!HN($IDHN$, $IDSN$, $pkHN$, $skSN$, $pkSN$, K)|
!UE($IDHN$, $pkSN$, K, $SUPI$))

Proverif thực hiện đặc tả các thuộc tính bảo mật bằng câu lệnh "*query*" trong thuật toán 5. Các thuộc tính xác thực được chặn bắt bằng các xác nhận tương ứng, thể hiện mối quan hệ giữa các sự kiện dưới dạng "nếu một sự kiện nào đó đã được thực hiện trong giao thức, thì một sự kiện khác đã được thực hiện trước đó".

3. KẾT QUẢ VÀ THẢO LUẬN

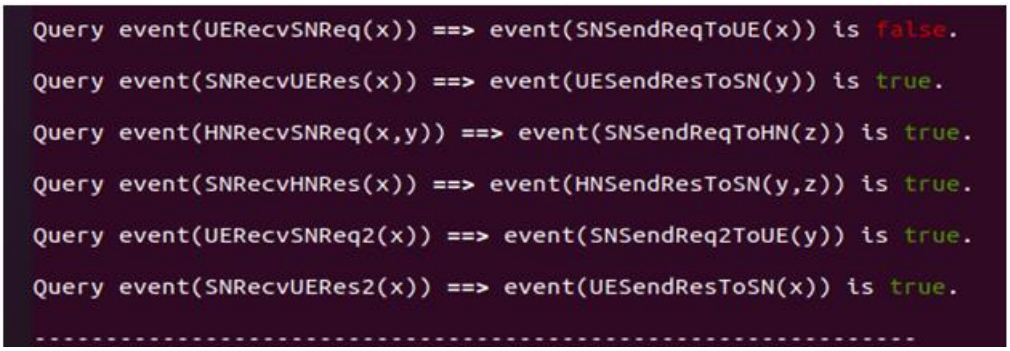
Giao thức 5G-AKA dựa trên blockchain sau khi được mô hình hóa và biên dịch qua công cụ proverif thu được kết quả kiểm tra các thuộc tính bảo mật của giao thức hình 6.

Kết quả hình 6 đã cho thấy các thuộc tính bảo mật như định danh vĩnh viễn của thuê bao (SUPI), các số ngẫu nhiên (R_2, R_3), khóa phiên chia sẻ giữa giao diện vô tuyến K_{SEAF}, khóa chia sẻ trước (K) giữa USIM và nhà mạng được bảo mật vì đầu ra của công cụ đều trả về kết quả "true". Điều này có nghĩa là kẻ tấn công không thu thập được thông tin gì trong quá trình giao thức diễn ra hay không có rò rỉ thông tin đối với những dữ liệu nhạy cảm này.



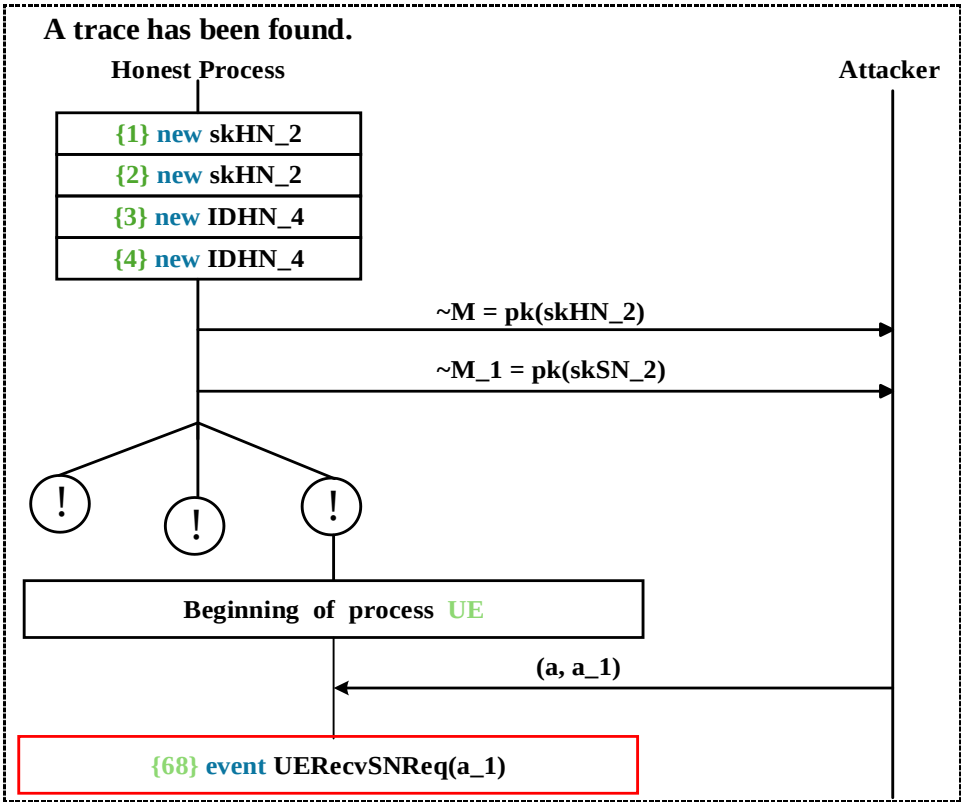
Hình 6. Kết quả kiểm tra các thuộc tính bảo mật.

Hơn nữa, đối với các thuộc tính xác thực của giao thức giao thức 5G-AKA dựa trên Blockchain, kết quả kiểm tra thể hiện trong hình 7.



Hình 7. Kết quả kiểm tra các sự kiện.

Kết quả hình 7 cho thấy, tất cả các thuộc tính xác thực, ngoại trừ cái đầu tiên thì đều được thỏa mãn bởi giao thức. Bằng cách truy tìm kết quả kiểm tra của thông điệp đầu tiên chúng ta có thể thấy rằng bất kỳ kẻ tấn công nào cũng có thể hoạt động như một SN bằng cách tạo một thông báo cho giai đoạn đầu tiên của giao thức hình 8. Theo hình 8, Honest Process thể hiện cho các tiến trình trung thực với các giá trị skHN_2, IDHN_4 và skSN_2, IDSN_4 lần lượt tương ứng là các khóa riêng và định danh của HN và SN. Hai giá trị M và M_1 là các khóa công khai tương ứng SN và HN được tính toán từ các khóa riêng áp dụng hàm pk(). Attacker đại diện cho tiến trình của kẻ tấn công có thể tìm thấy một truy vết tấn công vào quá trình khi SN gửi cho UE qua sự kiện *event UERecvSNReq(a_1)*. Sự kiện này ghi lại khi chạy công cụ proverif minh chứng UE đã nhận yêu cầu từ SN với nội dung a_1. Tham chiếu trong hình 8 kết hợp với hình 2, (a_1) là dữ liệu R_1, ID_{SN}. Theo trace này sẽ thấy rằng UE không thể phân biệt được giữa SN thật và SN giả mạo vì khi nhận được thông tin này, UE vẫn tiếp tục tính toán và UI_c và SUCI để gửi lại SN ở bước 2 hình 2. Hơn nữa, các thông báo này được gửi trên kênh công khai nên kẻ tấn công có thể chặn bắt thông tin SUCI và sử dụng thông báo này để thực hiện tấn công phát lại (tức là, gửi lại thông báo req-id, ID_{SN} và SUCI) tới HN. Tuy nhiên, thông điệp này trước khi gửi đến HN sẽ được SC kiểm tra tính mới của yêu cầu, bằng cách tìm kiếm các bản ghi xác thực với req-id nếu thấy yêu cầu bị trùng lặp thì giao thức bị dừng lại. Qua kết quả này có thể thấy luôn tồn tại SN độc hại có thể hoạt động qua giao diện vô tuyến (vì nó là kênh công khai) nhưng dù SN độc hại tồn tại nó cũng không lấy được thông tin gì trong giao thức vì khóa K_{SEAF} và định danh SUPI của UE chỉ được tiết lộ cho SN thật (chỉ có SN thật mới có khóa bí mật sk_{SN} để giải mã thông điệp chứa SUPI và K_{SEAF}) ở bước cuối của giao thức khi tất cả các bên đã được xác thực. Điều này, khẳng định liên lạc giữa SN và HN sử dụng blockchain không chỉ giúp giải phóng giả định kênh liên lạc giữa SN và HN mà còn giúp ngăn chặn các tấn công phát lại, ngăn chặn được SN giả mạo hoạt động.



Hình 8. Truy vết khi kết quả đầu ra là false.

Điểm khác của bài báo so với [14] là đi tìm hiểu cụ thể về cách thức kiểm tra giao thức an toàn sử dụng công cụ proverif và phân tích làm rõ kết quả bên trong đặc biệt khi công cụ trả kết quả về "false" đã đưa ra được đồ hình tấn công hình 8 để chứng minh an toàn của giao thức trước các cuộc tấn công phát lại và giả mạo SN. Điều này, giúp các nhà nghiên cứu có cái nhìn toàn diện về cách thức kiểm tra giao thức mật mã bằng cách sử dụng mô hình tượng trưng cụ thể là proverif.

4. KẾT LUẬN

Bài báo này nhóm tác giả đã trình bày được phương pháp kiểm tra giao thức mật mã sử dụng công cụ proverif với việc phân tích kiểm tra cấu trúc, ngôn ngữ của công cụ là sử dụng tính toán pi và tự động phiên dịch thành mệnh đề horn để phân tích các đặc điểm bảo mật của giao thức. Trên cơ sở đó, bài báo ứng dụng mô phỏng kiểm tra giao thức 5G-AKA dựa vào blockchain để kiểm tra các tham số như: định danh vĩnh viễn của thuê bao, các số ngẫu nhiên, khóa chia sẻ giữa giao diện vô tuyến, khóa chia sẻ trước giữa USIM và nhà mạng được bảo mật. Với kết quả này góp phần giúp các nhà nghiên cứu và thiết kế giao thức mật mã có những cách tiếp cận phân tích giao thức mật mã là an toàn trước khi thử nghiệm và triển khai trong thực tế.

LỜI CẢM ƠN

Nhóm tác giả xin chân thành cảm ơn Học viện Kỹ thuật mật đã tài trợ cho nghiên cứu với mã số 14/2025/CS.

TÀI LIỆU THAM KHẢO

[1]. A. Peltonen, Formal Verification and Standardization of Security Protocols: Proverif and Extensions, 8th Internetnational Conference ICAIS on Artificial Intelligence and Security, Qinghai, China, July 15-20, 2022. https://doi.org/10.1007/978-3-031-06788-4_42

[2]. B.Blanchet, Modeling and Verifying Security Protocols with the Applied Pi Calculus and ProVerif, Foundations and Trends® in Privacy and Security, 1 (2016) 1–135. http://dx.doi.org/10.1561/3300000004.

[3].A.Armando, D.Basin, Y.Boichut, Y.Chevalier, L.Compagna, J.Cuellar, P.H.Drielsma, P.C.Heám, O.Kouchnarenko, J.Mantovani, S.Modersheim, D.V.Oheimb, M.Rusinowitch, J.Santiago, M.Turuani, L.Vigano, L.Vigneron, The AVISPA Tool for the Automated Validation of Internet Security Protocols and Applications, International Conference on Computer Aided Verification, Berlin, Heidelberg, Springer, 3576 (2005) 281–285. https://doi.org/10.1007/11513988_27

[4]. G. Lowe, Breaking and fixing the Needham-Schroeder Public-Key Protocol using FDR, International Workshop on Tools and Algorithms for the Construction and Analysis of Systems TACAS, Berlin, Heidelberg, Springer, 1055 (1996) 147–166. https://doi.org/ 10.1007/3-540-61042-1_43

[5].C.Cremers, Unbounded verification, falsification, and characterization of security protocols by pattern refinement, Proceedings of the ACM Conference on Computer and Communications Security, (2008) 119–128. https://doi.org/10.1145/1455770.1455787

[6].B. Schmidt, S. Meier, C. Cremers, D. Basin, Automated Analysis of Diffie-Hellman Protocols and Advanced Security Properties, in IEEE 25th Computer Security Foundations Symposium, (2012) 78–94. https://doi.org/10.1109/CSF.2012.25

[7].B. Blanchet, B. Smyth, V. Cheval, M. Sylvestre, Proverif 2.05: Automatic Cryptographic Protocol Verifier, User Manual and Tutorial, (2023) https://bblanche.gitlabpages.inria.fr/proverif/manual.pdf

[8]. B.Blanchet, The Security Protocol Verifier ProVerif and its Horn Clause Resolution Algorithm, In Proceedings HCVS/VPT, 373 (2022) 14–22. https://doi.org/10.48550/arXiv.2211.12227

[9]. B.Blanchet, Using Horn Clauses for Analyzing Security Protocols, Cryptology and Information Security Series, 5 (2011) 86-111. https://doi.org/10.3233/978-1-60750-714-7-86

[10]. K.Prakash, Balachandra, Authentication and Key Agreement in 3GPP Networks, Computer Science & Information Technology (CS & IT), (2015) 143-154. https://airccj.org/10.5121/csit.2015.51313

[11]. A.Braeken, M.Liyanage, J. M. P.Kumar, Novel 5G Authentication Protocol to Improve the Resistance Against Active Attacks and Malicious Serving Networks, in IEEE Access, 7 (2019) 64040-64052. https://doi.org/10.1109/ACCESS.2019.2914941

[12]. H.Yang, H.Zheng, J.Zhang, Y.Wu, Y.Lee, Y.Ji, Blockchain-based trusted authentication in cloud radio over fiber network for 5G, 16th International Conference on Optical Communications and Networks (ICOCN), Wuzhen, China, (2017) 1-3. https://doi.org/10.1109/ICOCN.2017.8121598

[13]. J. Xu, K. Xue, H. Tian, J. Hong, D. S. L. Wei, P. Hong, An Identity Management and Authentication Scheme Based on Redactable Blockchain for Mobile Networks, IEEE Transactions on Vehicular Technology, 69 (2020) 6688–6698. https://doi.org/10.1109/TVT.2020.2986041

[14]. M. Hojjati, A. Shafieinejad, H. Yanikomeroglu, A Blockchain-Based Authentication and Key Agreement (AKA) Protocol for 5G Networks, IEEE Access, 8 (2020) 216461–216476. https://doi.org/10.1109/ACCESS.2020.3041710