JTE HCMUTE
ISSN: 2615-9740

JOURNAL OF TECHNICAL EDUCATION SCIENCE
Ho Chi Minh City University of Technology and Education
Website: https://jte.edu.vn
Email: jte@hcmute.edu.vn

Advancing Security Protocol Verification: A Comparative Study of Scyther, Tamarin

Thi Minh Chau Le*, Xuan Thang Pham, Vinh Thinh Le
Ho Chi Minh City University of Technology and Education, Vietnam

*Corresponding author. *Email:* chaultm@hcmute.edu.vn

ARTICLE INFO

Received: 18/01/2024
Revised: 25/02/2024
Accepted: 26/02/2024
Published: 28/02/2024

KEYWORDS

Verification Protocol;
Falsification Protocol;
Analysis Protocol;
Scynther;
Tamarin.

ABSTRACT

In the dynamic field of Internet security, verifying and analyzing security protocols is crucial for ensuring secure and effective communication. This paper presents an analysis of leading tools used for the verification, falsification, and analysis of security protocols, focusing particularly on Scyther and Tamarin. We examine the methodologies, capabilities, and applications of these tools in various contexts, including cloud computing and IoT, highlighting their unique approaches and contributions to the field. The paper starts with an examination of the Scyther tool, emphasizing its innovative approach to automated falsification and verification, and its application in multi-protocol analysis. We then transition to exploring the use of Scyther in verifying key agreement protocols in cloud computing. A comparative analysis of Scyther and Tamarin is also presented, shedding light on their effectiveness in identifying security properties and revealing the strengths and weaknesses of different protocols. This is further enriched by a detailed look at the unbounded verification capabilities of Scyther. Additionally, the paper covers the capabilities of the Tamarin prover, exploring its advanced features in symbolic analysis, its ability to handle complex security properties, and its application in verifying protocol implementations. Through this paper, we aim to provide a comprehensive overview of the current landscape in security protocol verification, offering insights into the methodologies, challenges, and future directions in this essential area of research.

Doi: https://doi.org/10.54644/jte.2024.1523

1. Introduction

In the realm of digital security, the role of security protocols, especially key agreement protocols, is paramount. These protocols are the cornerstone of secure communication in an increasingly interconnected world, where data transmission often occurs over potentially insecure networks [1] [2]. Central to cryptographic key management systems, these protocols facilitate entities in establishing shared secret keys, which are essential for safeguarding subsequent communications. The authors in [3] investigate into the analysis and verification of hierarchical identity-based authenticated key agreement (HIB-AKA) protocols, employing advanced tools like Scyther and Tamarin for a comprehensive evaluation.

The landscape of security protocol research is diverse and evolving. Studies exploring identity-based authenticated key agreement protocols within the Diffie-Hellman family, particularly those enabled by Weil or Tate pairings, have shed light on intricate issues related to protocol design and security [4]. In the wake of emerging quantum computing threats, research focusing on lightweight lattice-based secure systems for authenticated key agreement protocols has gained prominence. These studies underscore the urgency for protocols that are not only secure but also efficient in the post-quantum era [5].

The Scyther tool [6] [7] [8] has emerged as a pivotal resource in the verification of security protocols. Its comprehensive approach to analyzing and falsifying security protocols has been instrumental in advancing our understanding of protocol vulnerabilities and strengths. In parallel, the Tamarin prover [9] [10] has been recognized for its robust capabilities in the symbolic analysis of security protocols. Its

application extends to a variety of complex protocol scenarios, demonstrating its versatility and depth in handling intricate security properties [11].

The introduction of Tamarin represents a significant leap in the field of security protocol verification. This novel framework pushes the boundaries of automation in state-of-the-art verification approaches, offering new perspectives and methodologies in protocol analysis. Furthermore, the Tamarin prover's ability to formalize a broad spectrum of protocols, adversary models, and properties, particularly in large-scale, real-world scenarios, marks a milestone in cryptographic protocol verification [12].

A notable advancement in the field is the development of a method that synergizes model learning technology with the Tamarin tool. This approach, aimed at verifying the security properties of finite-state machine (FSM) models derived from protocol implementations, addresses critical challenges in detecting logical errors. It marks a departure from traditional reliance on expert experience for compliance rule extraction, paving the way for more streamlined and accurate protocol verification processes [9].

This paper is structured to provide a comprehensive exploration of these themes. Section 2 presents a theoretical comparison, introducing essential verification tools such as Scyther and Tamarin, and discusses their significance in the contemporary security landscape. Section 3 details the directions and emerging trends of security tools and suggests directions for future research in the field of security protocol verification. Finally, Section 4 concludes the paper, summarizing the key findings, discussing the implications of our research.

2. Scyther, Tamarin: the comparison

Transitioning from the introduction to this section, we now focus on the foundational aspects of our study. This session is divided into three essential subsections, each dedicated to a significant tool in the field of security protocol analysis: the Scyther Tool and the Tamarin Prover. The first subsection is centered on the Scyther Tool, a key instrument in the verification and analysis of security protocols. We will explore its methodology, distinctive features, and its varied applications in both industrial and academic settings. Additionally, we will discuss its strengths and limitations to provide a comprehensive understanding of the tool's impact in the realm of security protocol analysis. Following our exploration of the Scyther Tool, the subsequent subsection will focus on the Tamarin Prover. We will examine Tamarin's unique approach to the symbolic analysis of security protocols, its capabilities, and its diverse applications. This analysis aims to shed light on how the Tamarin Prover contributes to and advances the verification and analysis of security protocols. Finally, the comparison between these tools have been draw with the detail of pros and cons.

A. The Scyther Tool

Detailed Analysis of Scyther's Methodology and Features.

The Scyther tool represents a significant advancement in the field of security protocol analysis. Developed with a focus on user-friendliness and efficiency, Scyther employs a unique methodology that sets it apart from other verification tools. Its core is based on a pattern refinement algorithm, which allows for the concise representation of (infinite) sets of execution traces. This algorithm is instrumental in assisting the analysis of classes of attacks and possible protocol behaviors, or in proving correctness for an unbounded number of protocol sessions.

Scyther's methodology is grounded in the Dolev-Yao intruder model, a widely accepted approach in security protocol analysis. This model assumes that the intruder has complete control over the network but cannot break cryptographic primitives. Scyther leverages this model to simulate potential attacks and identify vulnerabilities within the protocols. The tool's ability to handle unbounded verification with guaranteed termination is a notable feature, enabling it to provide conclusive results on the security of protocols under analysis.

Case Studies and Applications in Various Contexts.

Scyther has been applied successfully in a variety of contexts, demonstrating its versatility and effectiveness. It has been used in both academic research and industry to analyze and verify the security

of numerous protocols. For instance, Scyther has been instrumental in the analysis of industrial security protocols like IKE (versions 1 and 2) and ISO/IEC 9798. These case studies have shown Scyther's capability to handle complex protocols and provide valuable insights into their security properties. In academic settings, Scyther has been used as a teaching tool, helping students understand the intricacies of security protocols and their analysis. Its intuitive interface and graphical representations of protocol interactions make it an effective educational tool.

Strengths and Limitations.

One of Scyther's primary strengths is its user-friendly interface, which lowers the barrier to entry for those new to security protocol analysis. Its automated nature allows for quick and efficient analysis, saving time and resources. Additionally, Scyther's ability to handle unbounded verification and provide conclusive results adds to its reliability and effectiveness. However, Scyther is not without limitations. While it excels in the analysis of certain types of security properties, such as authentication and secrecy, it may not be as effective in dealing with more complex properties like non-repudiation or certain types of denial-of-service attacks. Furthermore, its reliance on the Dolev-Yao model, while beneficial in many cases, means that it may not account for certain real-world considerations, such as physical attacks or side-channel attacks. In short, the Scyther tool stands out as a powerful and user-friendly tool for the analysis of security protocols. The methodology, applications, strengths, and weaknesses of this approach underscore its importance in the domain of security protocol analysis, rendering it an invaluable resource for researchers, security professionals, and educators alike. By providing a comprehensive and automated approach to protocol verification, Scyther contributes significantly to the understanding and strengthening of security protocols in various digital communication environments. As with any tool, its effectiveness is maximized when users are aware of its scope and limitations, ensuring that it is applied appropriately within the context of a broader security analysis strategy. The following diagram illustrates the key aspects of the Scyther Tool, including its methodology, applications, strengths, and limitations. It provides a clear representation of how the tool functions and is utilized in various contexts.

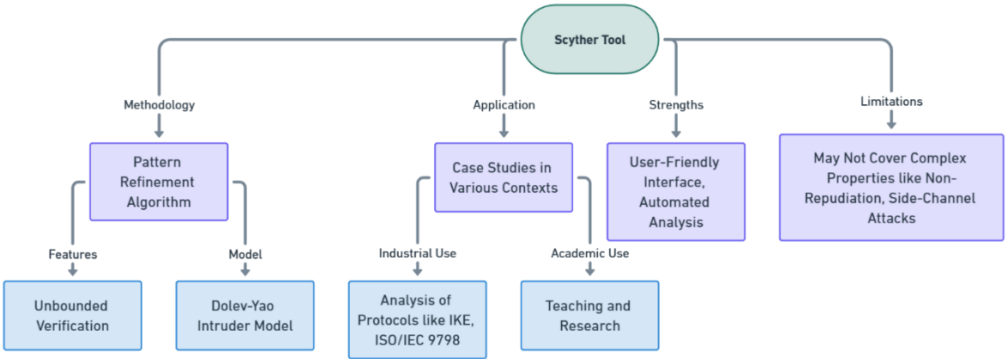

Figure 1. *Overrall key aspects of Scyther.*

Language.

The Scyther tool uses a specialized language for specifying security protocols, known as the Scyther Specification Language (SSL). This language is designed to be simple yet expressive enough to define various aspects of security protocols, including roles, messages, and cryptographic operations.

Key Features of SSL:

1. Role-Based Specification: Protocols are defined in terms of roles (like initiator, responder) and their interactions.
2. Message Format: Messages are defined using a straightforward syntax that describes the composition of messages sent and received.
3. Cryptographic Constructs: Common cryptographic operations like encryption, decryption, and hashing are supported.

4. Variables and Constants: The language allows the use of variables (for dynamic values like nonces) and constants (for static values like keys).

To be clearer, considering an example is of the Needham-Schroeder Public Key Protocol using the Scyther Specification Language (SSL). This protocol is designed for establishing a secure communication channel between two parties, identified as **I** (Initiator) and **R** (Responder), by exchanging nonces (random numbers) to verify each other's identity and establish shared secrets. Let's break down the example:

Protocol Definition:

- protocol ns3(I, R) {...}: Defines a new protocol named ns3 with two roles, I (Initiator) and R (Responder).

Role **I** (Initiator):

1. fresh ni: Nonce;: Initiator generates a fresh nonce ni.
2. var nr: Nonce;: Declares a variable nr to store the nonce received from the Responder.
3. send_1(I, R, {ni, I}pk(R));: Initiator sends a message to Responder, containing the nonce ni and its identity I, encrypted with Responder's public key pk(R).
4. recv_2(R, I, {ni, nr}pk(I));: Initiator receives a message from Responder, containing the original nonce ni and Responder's nonce nr, encrypted with Initiator's public key pk(I).
5. send_3(I, R, {nr}pk(R));: Initiator sends back the nonce nr received from Responder, encrypted with Responder's public key pk(R).

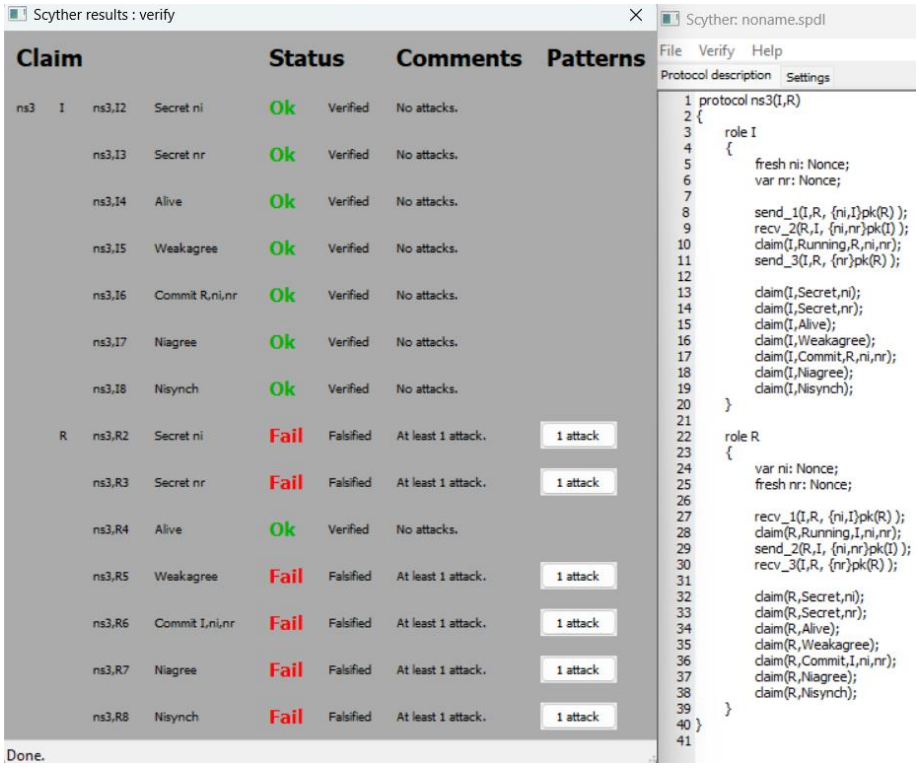

Figure 2. *Scyther laguage and input/output*

The Initiator then makes several claims about the protocol's security properties, such as secrecy of nonces, aliveness, agreement, commitment, and synchronization.

Role **R** (Responder):

1. var ni: Nonce;: Responder declares a variable ni to store the nonce received from the Initiator.
2. fresh nr: Nonce;: Responder generates a fresh nonce nr.
3. recv_1(I, R, {ni, I}pk(R));: Responder receives the first message from Initiator, decrypts it to get Initiator's nonce ni and identity I.

4. send_2(R, I, {ni, nr}pk(I));: Responder sends a message back to Initiator, containing the received nonce ni and its own nonce nr, encrypted with Initiator's public key pk(I).
5. recv_3(I, R, {nr}pk(R));: Responder receives the final message from Initiator, containing the nonce nr.

The Responder also makes similar claims about the protocol's security properties.

Security Claims:

- claim(...);: These statements are used to assert various security properties like secrecy (nonces are not known to others), aliveness (other party is active), agreement (both parties agree on the nonces), commitment (a party is committed to a session), and synchronization (nonces are synchronized between parties).

This protocol aims to securely establish a shared secret between the Initiator and Responder, ensuring that both parties are who they claim to be. The claims are used to verify the security properties of the protocol using the Scyther tool. The result is presented in Figure 2 that shows the number of attacks can affect the protocol.

Furthermore, Scyther can trace attacks by presenting graphs, as demonstrated in Figure 3

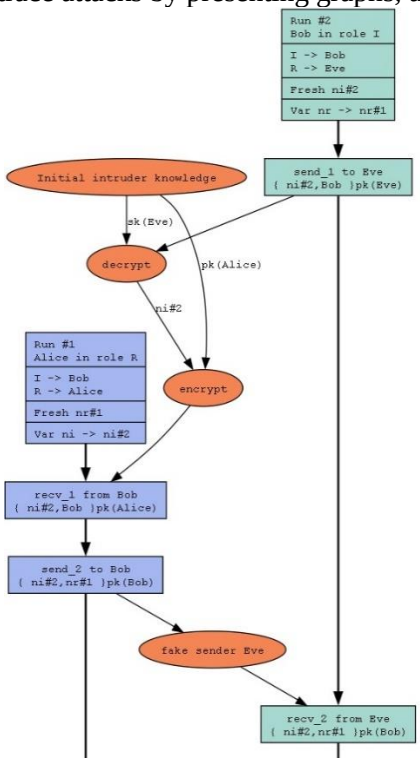

Figure 3. *Attack claims in Scyther*

B. The Tamarin Prover

Capabilities and Design. The Tamarin Prover is a sophisticated tool for security protocol verification, supporting both falsification and unbounded verification in the symbolic model. It employs multiset rewriting systems for protocol specification, defining a labeled transition system. The state of this system includes a symbolic representation of the adversary's knowledge, network messages, information about freshly generated values, and the protocol's state. Tamarin also supports the equational specification of cryptographic operators, such as Diffie-Hellman exponentiation and bilinear pairings, enhancing its capability to model complex cryptographic scenarios.[10]

Case Studies and Applications. Tamarin's versatility is evident in its application across various contexts. It has been effectively used to model and analyze web applications, with its repository offering numerous examples from diverse papers, serving as a valuable resource for modeling other protocols. Its application in the symbolic analysis of security protocols further underscores its utility in practical scenarios.

Strengths and Limitations. Tamarin excels in handling protocols with non-monotonic mutable global state and complex control flows, such as loops. It supports a form of induction and efficiently parallelizes its proof search, making it a powerful tool for protocol analysis. However, its operation within the symbolic model means it can only capture attacks that fall within this model and a given equational theory. Additionally, due to the undecidable nature of the underlying verification problems, Tamarin is not guaranteed to terminate.

Notable Applications and Use Cases. Tamarin has been instrumental in discovering new attacks, significantly impacting several real-world deployments. Its formal analysis of 5G protocols and symbolic analysis of web single-sign-on protocols are notable examples of its application in critical and contemporary technological domains. Several studies and surveys have focused on Tamarin, contributing to our understanding of its capabilities and applications. For instance, a paper [12] provides a comprehensive overview of Tamarin and surveys some of the significant results achieved with it. Another paper, [10] offers an in-depth look at the tool and its various applications. In short, the Tamarin Prover stands as a robust and versatile tool in the field of security protocol verification. Its advanced capabilities, coupled with its application in a range of contexts, from web applications to 5G protocols, demonstrate its critical role in enhancing the security of digital communications. While its strengths in handling complex protocols and cryptographic operators are notable, awareness of its limitations, particularly in the symbolic model and the potential for non-termination, is essential for its effective application. The body of research and case studies surrounding Tamarin not only underscores its importance but also provides a rich resource for further exploration and development in the field of security protocol analysis. The following diagram summarizes the key aspects of the Tamarin Prover, including its capabilities, applications, strengths, limitations, and notable use cases. It provides a clear representation of how the Tamarin Prover functions and is utilized in various contexts, highlighting its versatility and impact in the field of security protocol analysis.

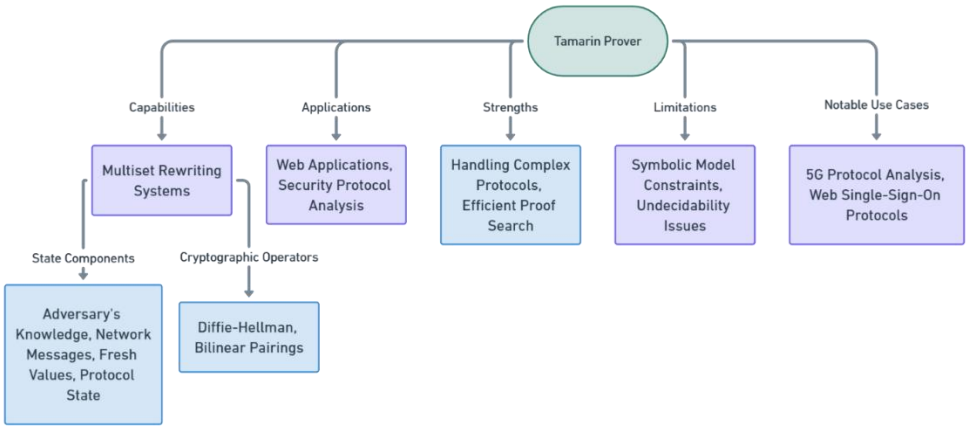

Figure 4. *Overrall key aspects of Tamarin*

Language. The Tamarin Prover uses a specialized language based on multiset rewriting rules for specifying and analyzing security protocols. This language is designed to express the behavior of protocols in terms of actions (like sending and receiving messages) and states. It is quite expressive and allows for the specification of complex properties and behaviors of security protocols.

Key Features of Tamarin's Language:

1. Multiset Rewriting Rules: These rules define how the state of the protocol evolves with each action.
2. State Representation: The state includes the knowledge of the adversary, the messages on the network, and the internal state of the protocol participants.
3. Fact-Based Syntax: The language uses facts to represent the state of the system and the actions taken by the protocol participants.
4. Support for Cryptographic Primitives: Tamarin can model various cryptographic operations and equational theories.

JTE HCMUTE
ISSN: 2615-9740
JOURNAL OF TECHNICAL EDUCATION SCIENCE
Ho Chi Minh City University of Technology and Education
Website: https://jte.edu.vn
Email: jte@hcmute.edu.vn

Let's consider a simple example of a protocol with two roles: an initiator and a responder. The protocol involves the initiator sending a nonce to the responder, who then responds with the nonce encrypted with a shared key.

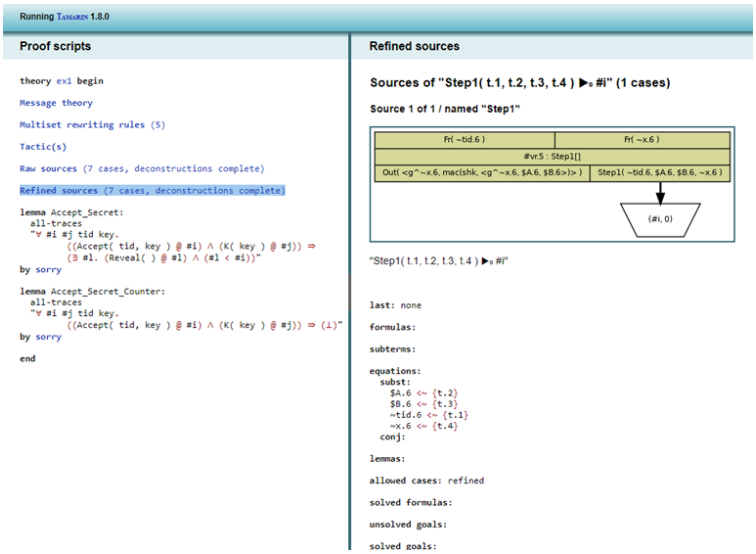

Figure 5. *Tamarin Proper*

In this Tamarin representation:

- Each **rule** represents an action in the protocol (sending or receiving a message).
- The **[...]** brackets represent the multiset before and after the action.
- **!Initiator(I)** and **!Responder(R)** denote the roles of the initiator and responder.
- **Fr(~ni)** denotes the generation of a fresh nonce **ni**.
- **In(...)** and **Out(...)** represent incoming and outgoing messages.
- **St_I** and **St_R** are states in the protocol.
- **--[...]** → denotes the transition from one state to another, with labels representing events in the protocol.

In real-world protocols would typically be more complex, involving multiple steps and various cryptographic operations. The Tamarin Prover allows for the analysis of these protocols by checking them against various security properties like secrecy and authentication. The Needham-Schroeder Public Key Protocol can be explained in Tamarin Prover as the following diagram.

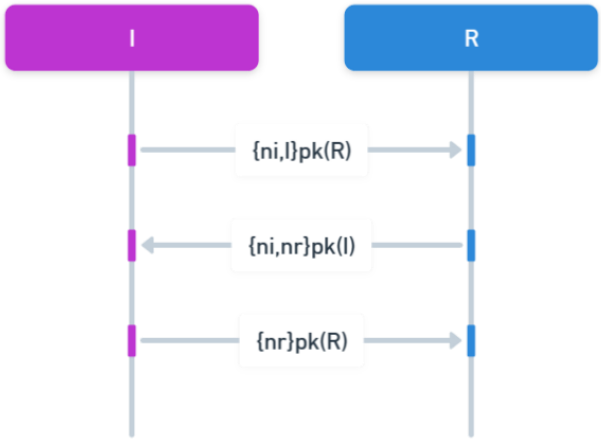

Figure 6. *Needham-Schroeder (NS) Public Key Protocol*

This diagram depicts the sequence of messages exchanged between the Initiator (I) and the Responder (R) in the NS protocol:

1. **I →R**: The Initiator sends its nonce **ni** and identity **I**, encrypted with the Responder's public key **pk(R)**.
2. **R → I**: The Responder replies with the Initiator's nonce **ni** and its own nonce **nr**, encrypted with the Initiator's public key **pk(I)**.
3. **I → R**: The Initiator confirms by sending the Responder's nonce **nr** encrypted with the Responder's public key **pk(R)**.

C. The Comparison

The following table presents a detailed comparative analysis of two widely used security protocol verification tools: Scyther and Tamarin. This comparison is based on various parameters such as their main purpose, protocol verification capabilities, protocol specification methods, and additional features.

Table 1. *The Comparison between Scythe and Tamarin Proper*

Aspect / Feature	Scyther (Pros)	Scyther (Cons)	Tamarin (Pros)	Tamarin (Cons)
Main Purpose	Ideal for automatic verification of security protocols, especially standard analyses.	May not offer the advanced features needed for complex protocol analysis scenarios.	Designed for advanced, cutting-edge features in security protocol verification.	Might be overly complex for basic or standard protocol analysis needs.
Protocol Verification	Excellent for scenarios with an unbounded number of sessions and nonces.	Limited in handling more complex protocol structures compared to Tamarin.	Superior in handling dynamic corruption and user-specified adversaries.	Requires more in-depth understanding of protocol complexities.
Protocol Specification	Linear role scripts make it user-friendly and easier to understand.	Less flexible in protocol specification compared to Tamarin's Multiset Rewriting.	Multiset Rewriting allows for more detailed and complex protocol specification.	Higher learning curve due to the complexity of Multiset Rewriting.
Analysis Features	Efficient characterizations of protocols, with a finite representation of behaviors.	May lack advanced analysis features like proof visualization.	Advanced features like attack finding, visualization, and API support for global state.	Could be overkill for simpler analysis requirements.
Cryptographic Message Model	Simpler, more straightforward free term algebra.	Less extensive than Tamarin's model, lacking advanced cryptographic supports.	Extensive model including Diffie-Hellman, bilinear pairing, and advanced cryptographic terms.	The complexity of the cryptographic message model can be a barrier for less experienced users.
User-friendliness	More accessible for beginners and for educational purposes.	Might not cater to needs of advanced users seeking sophisticated analysis tools.	Interactive proof construction appeals to advanced users.	Potentially intimidating for beginners or those seeking straightforward tools.
Efficiency	Highly efficient for standard protocol analyses.	May not perform as well in complex or advanced analysis scenarios.	Ideal for complex protocol verifications due to its advanced capabilities.	May require more resources and time for analysis compared to Scyther.

Educational Use	Excellent resources for learning and teaching about security protocols.	May not cover advanced topics in depth.	Provides detailed tutorial materials and manual for advanced learning.	Might be challenging for those new to security protocol analysis.
Development and Community Support	Active development and available on GitHub; good community support.	Limited to specific use cases and might not evolve rapidly.	Continuous development with cutting-edge updates; strong community involvement.	The tool's complexity could limit its accessibility and user base.

3. Results and Discussion

As we look towards the future of tools like Scyther and Tamarin, it's clear that the field of security protocol analysis is poised for significant evolution. This evolution is driven by rapid technological advancements, changing security landscapes, and the ever-growing complexity of digital communication networks. The potential future directions and emerging trends for these tools can be discussed through several key lenses.

Adapting to Technological Shifts.

The integration of Scyther and Tamarin with emerging technologies such as quantum computing and artificial intelligence represents a frontier for exploration. These technologies could revolutionize how security protocols are analyzed, bringing in new capabilities like predictive modeling and enhanced attack simulation. As these technologies mature, they could offer unprecedented analytical power, enabling these tools to tackle more complex and sophisticated security challenges.

Evolving with Cryptographic Developments.

Cryptography is at the heart of security protocol analysis. As new cryptographic methods emerge, particularly in the realm of quantum-resistant algorithms, both Scyther and Tamarin will need to evolve. The ability to support and analyze these new algorithms will be crucial for maintaining relevance in the face of potential quantum computing threats.

Enhancing User Experience and Accessibility.

A critical aspect of the future development of these tools involves making them more accessible and user-friendly. Improvements in user interface design and educational resources could lower the barrier to entry, allowing a broader range of users to engage with these tools effectively. This democratization of access is key to fostering a wider understanding and application of security protocol analysis.

Leveraging Cloud and Distributed Computing.

The power of cloud-based and distributed computing opens up new avenues for handling complex analyses more efficiently. By utilizing these technologies, Scyther and Tamarin could offer enhanced performance and scalability, allowing for more extensive and thorough protocol analyses.

Specialization for IoT and Emerging Networks.

With the Internet of Things (IoT) becoming increasingly ubiquitous, there's a growing need for tools like Scyther and Tamarin to specialize in analyzing IoT-specific security protocols. This specialization would ensure that the unique challenges and configurations of IoT networks are adequately addressed.

Interoperability and Collaboration.

The future of security tools lies in their ability to work in concert with other systems and frameworks. Enhancing interoperability would allow Scyther and Tamarin to become integral parts of a larger security analysis ecosystem, thereby increasing their utility and applicability.

Addressing New Security Threats.

As the landscape of cyber threats evolves, these tools must also adapt to address emerging security challenges. This continual adaptation is necessary to ensure that they remain effective in identifying and mitigating novel types of attacks.

Fostering Community and Open-Source Development.

The role of community contributions in the development of these tools cannot be overstated. An active open-source community can drive innovation and keep the tools updated with the latest trends and best practices in the field.

Educational Initiatives.

Finally, the importance of education and training in the use of these tools is paramount. Developing comprehensive educational resources will help cultivate a new generation of users proficient in security protocol analysis.

In short, the future directions and emerging trends for Scyther and Tamarin reflect a broader trajectory in the field of security protocol analysis. As these tools evolve, they will likely continue to play a pivotal role in shaping the landscape of digital security and privacy.

4. Conclusions

In conclusion, our comparative analysis and exploration of Scyther and Tamarin have provided significant insights into the current capabilities and future trajectories of these tools in the domain of security protocol verification.

Scyther, with its user-friendly approach and efficient processing for standard analyses, serves as a robust tool for those entering the field or dealing with less complex security protocols. Its strength lies in its simplicity and accessibility, making it an excellent resource for educational purposes. Tamarin, on the other hand, stands out for its advanced analysis capabilities, catering to more complex and detailed security protocol scenarios. It offers a broader range of features, including support for sophisticated cryptographic models and interactive proof construction, making it well-suited for in-depth research and high-level protocol analysis. The evolving landscape of digital security, marked by emerging technologies and novel cryptographic challenges, indicates a promising future for both tools. As security protocols become more complex, especially with the advent of IoT and quantum computing, the adaptability and advancement of tools like Scyther and Tamarin become crucial. Their continued development and integration with new technologies and cryptographic methods will be pivotal in addressing future security challenges. Moreover, the active involvement of the research and developer communities in enhancing these tools, through open-source contributions and educational initiatives, will play a significant role in their evolution. This collaboration is essential for keeping these tools at the forefront of security protocol verification. In essence, Scyther and Tamarin are not just tools but are representative of the dynamic and ever-evolving field of security protocol analysis. Their impact extends beyond mere verification; they are instrumental in shaping the methodologies and approaches in this critical area of cybersecurity research. As the digital world continues to grow in complexity and scale, the importance of advanced, versatile, and user-friendly tools for security protocol verification will only increase, making the contributions of Scyther and Tamarin more valuable than ever. Finally, our analysis serves as a valuable guide for researchers and practitioners in understanding the intricacies of these tools and their significant impact on the development and verification of secure communication protocols.

REFERENCES

[1] M. A. Fikri, K. Ramli, and D. Sudiana, “Formal Verification of the Authentication and Voice Communication Protocol Security on Device X Using Scyther Tool,” IOP Conf. Ser. Mater. Sci. Eng., vol. 1077, no. 1, p. 012057, Feb. 2021, doi: 10.1088/1757-899X/1077/1/012057.

[2] N. Dalal, J. Shah, K. Hisaria, and D. Jinwala, “A Comparative Analysis of Tools for Verification of Security Protocols,” Int. J. Commun. Netw. Syst. Sci., vol. 03, no. 10, pp. 779–787, 2010, doi: 10.4236/ijcns.2010.310104.

[3] H. A. Elbaz, M. H. Abd-elaziz, and T. M. Nazmy, “Analysis and Verification of a Key Agreement Protocol over Cloud Computing Using Scyther Tool,” vol. 2, no. 2, 2014.

[4] L. Chen and C. Kudla, “Identity based authenticated key agreement protocols from pairings,” in 16th IEEE Computer Security Foundations Workshop, 2003. Proceedings., Pacific Grove, CA, USA: IEEE Comput. Soc, 2003, pp. 219–233. doi: 10.1109/CSFW.2003.1212715.

[5] Y. Yang, H. Yuan, L. Yan, and Y. Ruan, “Post-quantum identity-based authenticated multiple key agreement protocol,” ETRI J., vol. 45, no. 6, pp. 1090–1102, Dec. 2023, doi: 10.4218/etrij.2022-0320.

[6] C. J. F. Cremers, "The Scyther Tool: Verification, Falsification, and Analysis of Security Protocols," in Computer Aided Verification, vol. 5123, A. Gupta and S. Malik, Eds., in Lecture Notes in Computer Science, vol. 5123. , Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 414–418. doi: 10.1007/978-3-540-70545-1_38.

[7] C. J. F. Cremers, "Unbounded verification, falsification, and characterization of security protocols by pattern refinement," in Proceedings of the 15th ACM conference on Computer and communications security, Alexandria Virginia USA: ACM, Oct. 2008, pp. 119–128. doi: 10.1145/1455770.1455787.

[8] C. Xi and L. Siqi, "Research on semantics and algorithm of formal analysis tool Scyther," in 2022 IEEE 4th International Conference on Civil Aviation Safety and Information Technology (ICCASIT), Dali, China: IEEE, Oct. 2022, pp. 1058–1074. doi: 10.1109/ICCASIT55263.2022.9987170.

[9] X. Zhang, Y. Zhu, C. Gu, and X. Miao, "A Formal Verification Method for Security Protocol Implementations Based on Model Learning and Tamarin," J. Phys. Conf. Ser., vol. 1871, no. 1, p. 012102, Apr. 2021, doi: 10.1088/1742-6596/1871/1/012102.

[10] S. Meier, B. Schmidt, C. Cremers, and D. Basin, "The TAMARIN Prover for the Symbolic Analysis of Security Protocols," in Computer Aided Verification, vol. 8044, N. Sharygina and H. Veith, Eds., in Lecture Notes in Computer Science, vol. 8044. , Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 696–701. doi: 10.1007/978-3-642-39799-8_48.

[11] Y. Xiong, C. Su, W. Huang, F. Miao, W. Wang, and H. Ouyang, "Verifying Security Protocols using Dynamic Strategies." arXiv, Aug. 25, 2019. Accessed: Jan. 18, 2024. [Online]. Available: http://arxiv.org/abs/1807.00669

[12] D. Basin, C. Cremers, J. Dreier, and R. Sasse, "Tamarin: Verification of Large-Scale, Real-World, Cryptographic Protocols," IEEE Secur. Priv., vol. 20, no. 3, pp. 24–32, May 2022, doi: 10.1109/MSEC.2022.3154689.

Le Thi Minh Chau graduated from university in Infomatics in 2005 at the Open University and received a master's degree in computer science in 2012 at Ho Chi Minh City University of Technology. She is currently working at the Faculty of Information Technology, HCMC University of Technology and Education, Viet Nam. Her research interests include security, privacy, Big Data AI, and related technologies. Email: chaultm@hcmute.edu.vn.

ORCID: https://orcid.org/0009-0004-8372-9098

Pham Xuan Thang graduated with a Bachelor's degree in Information Technology from Ho Chi Minh City University of Technology and Education in 2019. He has accumulated 6 years of work experience as a Ho Chi Minh City University of Technology and Education software specialist. His research interests encompass various topics within Information Technology, including software development, data analysis, and machine learning. His research interests include security, privacy, Big Data AI, and related technologies. Email: thangpx@hcmute.edu.vn.

Le Vinh Thinh completed a Ph.D. at the Conservatoire National des Arts et Métiers (CNAM), Paris, France, in 2017. He is currently working at the Faculty of Information Technology, HCMC University of Technology and Education, Viet Nam. He has been the author and co-author of over 20 peer-reviewed scientific articles. He has been actively involving in various research projects and collaborations both nationally and internationally. He is the dedicated educator, committed to fostering a conducive learning environment and advancing the field through research and innovation. His research interests include Trust and Reputation Systems, Security, Mobile Cloud Computing, and the Internet of Things (IoT).
Email: thinhlv@hcmute.edu.vn. ORCID: 0000-0001-5951-096X.