Fast and Furi-hash: An Improved General Framework for Automated Discovery of Hash-Based Protocol Attacks

Hafeez Muhammed and Arun Natarajan

***Abstract*—The *Hash Gone Bad* paper [1] highlights a critical gap in symbolic protocol verification: tools like ProVerif cannot natively model the associative properties of real-world hash constructions, relying instead on manual, bounded approximations. This leads to incomplete analysis and significant modeling overhead. Our project extends this work by proposing to replace the current recursive workaround with a sound, unification-based solver. The key contributions of this work are:**

- **Integration of a certified AC-unification algorithm into ProVerif to enable full associative reasoning.**
- **Improved modeling accuracy and completeness for discovering complex hash-based attacks.**
- **A significant step towards automating the analysis of protocols with complex algebraic properties.**

This research aims to enhance ProVerif's expressivity and reduce manual effort, making advanced, algebraically-aware protocol verification more accessible and robust.

***Index Terms*—ProVerif, Symbolic Verification, Hash Functions, AC-Unification, Protocol Analysis.**

I. Introduction

THE security of modern cryptographic protocols is critically dependent on the integrity of their underlying cryptographic primitives, particularly hash functions. While formal verification has become an essential tool for identifying protocol flaws, many analysis techniques rely on an idealized abstraction of these primitives. The Random Oracle Model (ROM), for instance, assumes a "perfect" hash function, free from the vulnerabilities such as collisions or length-extension attacks that affect many widely deployed algorithms like MD5 and SHA-1. This creates a significant gap between a protocol's proven security in an idealized model and its actual security in a real-world implementation.

The foundational work in "Hash Gone Bad" by Cheval et al. [1] directly addresses this gap by introducing a framework to model such real-world weaknesses in symbolic analysis tools. A key innovation was the extension of the ProVerif tool with recursive computation functions (compfun), a feature designed to approximate the associative properties of hash constructions like Merkle-Damgård, which standard ProVerif cannot natively handle. However, as noted by the authors themselves, this powerful solution introduces a new challenge: the complex compfun models must be written by hand, a difficult and error-prone process that limits the framework's broader applicability and was explicitly left as "future work" [1].

Instead of automating the generation of these bounded approximations, we propose to replace them entirely. Our project aims to integrate a sound, unification-based solver directly into ProVerif , based on the certified AC-unification algorithm from Ayala-Rincón et al. [2]. This approach targets the underlying limitation—ProVerif's lack of native associative reasoning—aiming to replace the bounded, manual workaround with a more complete and automated solution

This report details our progress. Section II provides an overview of the base paper [1] and its core contributions. Section III describes the empirical work undertaken to reproduce the original findings, including the challenges and solutions related to the Docker environment. Section IV analyzes the research gaps identified, leading to our proposed solution, which is detailed in Section V along with a comprehensive evaluation plan. Finally, Section VI discusses the expected outcomes, and Section VII concludes the report.

II. Literature Survey

This research is built upon two key areas of prior work: the modeling of hash function weaknesses in symbolic verifiers and the theoretical development of AC-unification.

A. Modeling Hash Weaknesses in ProVerif

The primary foundation for this project is the *Hash Gone Bad* (USENIX Security 2023) paper by Cheval et al. [1]. This work introduces a novel methodology and tool extensions to model hash function weaknesses in ProVerif and Tamarin. Key contributions include:

- A threat model lattice capturing different hash weaknesses.
- Introduction of recursive computation functions in ProVerif (`computation_function.ml`) enabling partial simulation of associative concatenation.
- Case studies on 20+ protocols demonstrating rediscovery of known and discovery of new hash-collision attacks.

However, as we note in Section IV, ProVerif's current approach uses bounded associativity (recursion limited by transcript arity) to guarantee termination, at the cost of incomplete associative reasoning. This manual, bounded approximation is the central problem we aim to solve.

B. Certified AC-Unification

In parallel, the challenge of reasoning about terms with associative properties has been a long-standing problem in automated reasoning. A significant recent breakthrough is the

work by Ayala-Rincón et al. [2], which provides a **certified first-order AC-unification algorithm**. This paper presents a sound and complete unification algorithm for Associative-Commutative (AC) theories and provides a formal, machine-checked proof of its correctness. This algorithm provides the practical, verified "solver" that was missing, making it a candidate to replace the bounded approximations used in [1].

III. Reproduction and Empirical Work

Attempting to reproduce the original experiments [1] involved:

- Running the original `proverif-compfun` Docker container [3], which was tightly coupled to Tamarin.
- Creating a new Dockerized environment to decouple ProVerif extensions, enabling independent testing.
- Using a custom script to perform a diff of original ProVerif 2.04 and the modified source to highlight key changes.

Through this setup we validated that the ProVerif extension and associated models run correctly in isolation. The repositories and tools are available here:

- Project repo: [4]
- Docker image: [5]

IV. Research Gaps

The "Hash Gone Bad" framework [1] highlights two significant research gaps. First, as acknowledged by its authors [1], the manual, recursive `compfun` models are a complex and bounded approximation of true associativity. Second, full associative unification remains unimplemented in ProVerif due to its theoretical and practical complexity.

The base paper [1] explicitly identifies automating the *generation* of their recursive models as "future work". However, this would only automate the creation of an *approximation*.

Our project targets the more fundamental gap: **the lack of a true associative solver**. We propose that by integrating the certified AC-unification algorithm from Ayala-Rincón et al. [2], we can *replace* the bounded, recursive workaround entirely. This directly addresses the core limitation noted in the base paper [1], moving beyond automating approximations to enabling full associative reasoning, which would significantly extend ProVerif's expressivity and reduce manual modeling effort.

V. Proposed Solution and Evaluation Plan

A. Proposed Solution

The proposed work seeks to overcome the limitations of bounded associative reasoning in ProVerif by integrating a certified AC-unification solver into its core computation engine. Specifically, we build upon the algorithm presented in *Certified First-Order AC-Unification and Applications* [2], which provides a sound and complete approach for reasoning over associative-commutative (AC) theories.

Our approach replaces ProVerif's existing recursive, bounded model used to approximate associative concatenation with a fully unification-based solver for associative terms. This integration allows ProVerif to evaluate term equality modulo associativity, thereby enabling complete associative reasoning rather than the limited, depth-bounded approach currently in use.

This integration moves beyond the semi-manual "recursive predicate" framework proposed in [1]. By automating this reasoning process through AC-unification, we aim to make ProVerif significantly more user-friendly and capable of modeling constructions without manual intervention.

B. Research & Development Plan

The implementation phase will begin with a prototype integration of the AC-unification algorithm within the `computation_function.ml` module of ProVerif's OCaml codebase. This step involves adapting the existing symbolic reasoning pipeline to invoke AC-unification whenever associative terms are encountered during verification.

Subsequently, we will design and evaluate a set of heuristics and bounds to preserve termination and control computational complexity. The extended framework will then be validated on representative case studies such as the Sigma protocol [1], which is known to involve associative concatenations. These models will serve to demonstrate both the correctness and scalability of the unification-based reasoning approach.

C. Evaluation Plan

Evaluation will focus on four dimensions:

- Correctness: Verify that the new model rediscovers existing known hash-collision attacks.
- Performance: Measure verification time, termination behavior, and memory usage compared to the bounded approach.
- Ablation Study: Vary unification bounds and heuristics to assess the trade-off between completeness and scalability.
- Robustness: Test on a set of protocols with complex concatenations.

VI. Expected Outcomes

Upon successful completion, this work will result in a generalized ProVerif extension that supports sound and automated associative unification. The enhanced tool is expected to substantially improve modeling accuracy and completeness in detecting protocol vulnerabilities related to hash-based constructions.

Moreover, this project will contribute to a deeper understanding of the trade-offs between expressivity and computational feasibility in symbolic reasoning with algebraic theories. All developed code, protocol models, and supporting documentation will be made publicly available through open-source repositories [4], [5], thereby facilitating reproducibility and future research in automated, algebraically-aware protocol verification.

VII. Conclusion

By addressing the limitations of bounded associativity [1] and moving towards full associative unification, this project aims to significantly enhance the expressivity and automation of symbolic protocol verification in ProVerif. The integration of a certified AC-unification algorithm [2] will enable more accurate modeling of real-world hash constructions, reduce manual effort, and potentially uncover new classes of protocol attacks. This work lays the foundation for future research in automated, algebraically-aware protocol verification.

Acknowledgment

We would like to express our sincere gratitude to our project advisor, Dr. Vinod Pathari, for his invaluable guidance, continuous support, and insightful feedback throughout the course of this research. His expertise was instrumental in shaping the direction of this project.

We also wish to thank the authors of "Hash Gone Bad," [1] whose foundational work provided the primary inspiration and basis for our research. Finally, we are grateful to the Department of Computer Science & Engineering at NIT Calicut for providing us with the resources and academic environment necessary to complete this work.

References

[1] V. Cheval, C. Cremers, A. Dax, L. Hirschi, C. Jacomme, and S. Kremer, "Hash Gone Bad: Automated Discovery of Protocol Attacks that Exploit Hash Function Weaknesses," in *Proc. 32nd USENIX Security Symposium*, 2023.

[2] M. Ayala-Rincón, M. Fernández, G. F. Silva, T. Kutsia, and D. Nantes-Sobrinho, "Certified First-Order AC-Unification and Applications," *Journal of Automated Reasoning*, vol. 68, no. 1, 2024.

[3] V. Cheval *et al.*, "Docker image and models for 'Hash Gone Bad'," GitHub Repository, 2023. [Online]. Available: https://github.com/charlie-j/symbolic-hash-models

[4] A. Natarajan and H. Muhammed, "Hash Gone Good," GitHub Repository, 2025. [Online]. Available: https://github.com/arunnats/hash-gone-good

[5] A. Natarajan, "ProVerif with Computation Functions," Docker Hub Repository, 2025. [Online]. Available: https://hub.docker.com/repository/docker/arunnats2004/proverif-compfun/general