Improving SHA-2 Collisions

Using Satisfiability

Modulo Theory (SMT) Solvers

BSc Computer Science  
Marcel Barlik  
[marcel.barlik@city.ac.uk](mailto:marcel.barlik@city.ac.uk)

# Table of Contents

[Table of Contents 1](#_Toc1)

[Abstract 2](#_Toc2)

[Introduction 2](#_Toc3)

[Research Questions 2](#_Toc4)

[Background Knowledge (8.2.4) 3](#_Toc5)

[Related Work 3](#_Toc6)

[Glossary 3](#_Toc7)

[Theoretical Framework 3](#_Toc8)

[Methodology (8.2.5) 3](#_Toc9)

[Benchmark Methodology 3](#_Toc10)

[Cryptanalytic Techniques 4](#_Toc11)

[Experimental Setup 4](#_Toc12)

[Results & Analysis (8.2.6) 5](#_Toc13)

[Security Implications 5](#_Toc14)

[Limitations and Future Work 5](#_Toc15)

[Conclusion 5](#_Toc16)

[References 5](#_Toc17)

[Appendices 5](#_Toc18)

Document Version: 1.0

|  |  |
| --- | --- |
| **Version** | **Date** |
| 1.0 | 2025-02-02 |

# Abstract

* Short findings from “Analysis” part

# Introduction

This research project investigates potential measurable quantifiable performance differences in SMT solvers and their parameters. This helps shape future research with the knowledge and understanding of the impacts of certain SAT solving lemmas on SHA2 collisions.

utilise unexplored opportunities that have arisen from advances in *“New Records in Collision Attacks on SHA-2”* (Li, Y. Liu, F. And Wang, G, 2024). This research will expand on the novel concept of using a Satisfiability Modulo Theory (SMT) solver for practical SHA-2 collisions, using the principles and mathematics described, in addition to their code as a reference.

* Problem Statement
* Contributions
* Paper Structure

## Research Questions

**RQ1:** Does using a more effective SMT solver yield better SHA-256 collision results?

**RQ2**: Can encodings provided in the research (Li, Y. Liu, F. And Wang, G, 2024) be improved upon, aiming for better practical SHA-256 collisions?

# Background Knowledge (8.2.4)

## Related Work

* Previous cryptanalysis approaches
* Theoretical foundations
* References to base paper
* Gaps in existing literature

## Glossary

* Glossary of words and terms (8.2.8)

# Theoretical Framework

* (Math Equations for RQ2)
* Mathematical foundations
* Cryptographic primitives
* Attack models

# Methodology (8.2.5)

* Research design
* Tools and Technologies

## Benchmark Methodology

* Data collection
* Design
* Evaluation

Slide 4 (https://moodle4.city.ac.uk/pluginfile.php/1093089/mod\_resource/content/6/UGP\_Session-Methods\_2024-25-v10.pptx.pdf):

What are the content and organisation of the Method Chapter?

What you need to report and in how much detail

– How do you report methods for your software development and

assessment?

– How do you report methods for experiments, measurements etc.?

– How do you report role of supervisor / client / others?

– How do you report use of AI aids?

– How do you report on Legal, Social, Ethical, Professional Issues in this

chapter?

– What goes into the Method chapter, what elsewhere?

Definition of how you ran your project: both your overall plan and choice of steps to go though, and how it turned out in practice

A justification of why you chose to do it this way: explaining your decisions, both for your initial planning and in response to events during the project

– product development lifecycle followed

– technologies for building your product

– methods for assessing it

– methods for any experiments or measurements

– how the work to be done was subdivided into phases

– timeline of the phases of work, specific incidents causing plan changes

# Cryptanalytic Techniques

* Detailed info of the approach taken
* Algorithms and methods developed
* GitHub reference (impl details/overview)

# Experimental Setup

* Hardware/Software env
* Reproducibility considerations
* Param settings
* Characteristics

# Results & Analysis (8.2.6)

* RQ1 - Benchmark results (graphs)
* RQ1 Discussion
* Performance analysis
* Comparison with previous approaches

# Security Implications

* Vulnerabilities identified (if any)
* Impact assessment

# Limitations and Future Work

* Difficulties while developing research
* Personal limitations (for e.g. Limited knowledge)
* Potential improvements
* Future research directions
* Pain points during the project

# Conclusion

* Summary of findings
* Significance of results
* Final remarks

# References

[1] <https://github.com/bitwuzla/bitwuzla/issues/169>

[2] <https://github.com/bitwuzla/bitwuzla/commit/fc8610fcfad12902d4eae4741ceb386524cf9b10>

@misc{cryptoeprint:2024/349,

author = {Yingxin Li and Fukang Liu and Gaoli Wang},

title = {New Records in Collision Attacks on {SHA}-2},

howpublished = {Cryptology {ePrint} Archive, Paper 2024/349},

year = {2024},

url = {https://eprint.iacr.org/2024/349}

}

# Appendices

* Detailed proofs
* Code snippets

(8.2.10) The following must be included:

* Appendix A: Project Definition Document;
* Appendix B: Reuse Summary (this must always be Appendix B; see Section 9.6, "Reusing software and how to document re-use");
* (mandatory for MSci students in Cybersecurity, good practice for all MSci students) Appendix C: Minutes of Supervisory Meetings (this must always be Appendix C);
* Complete records of each interview; questionnaires and complete questionnaire replies;
* Requirements;
* Routine design documentation;
* Source code, with instructions for building an executable version (as part of the “product package” submission; see next section);
* Test plans and test results; output listings; displays etc.;
* Any software installation guides and user guides produced (as part of the “product package” submission; see next section);
* An executable version of your software with installation instructions, or a URL if it is a web-based application that can be tested online (as part of the “product package” submission; see next section);
* All the reports produced for a client;
* Additional screen shots of your product running that are useful for understanding the product.

Other stuff to do:

* Reference PDD and what was delivered/wasn’t, what was improved on, etc
* must not exceed 12,000 words
* Same stuff as in 4.5.1 for cover sheet
* 8.2.2
* 8.2.3
* Video Submission
  + 15 minute demonstration video
    - Describe the project idea, its aims and how it met them
    - Describe main parts of the code
    - Describe how to use the project
    - Footage of the project running
    - Footage of the parts of code described
    - Voice over describing the content
* Talk about the representation they use being similar to what I found to be the most optimal.
* Talk about issue for found with Bitwuzla:
  + When experimenting with different solvers, I stumbled into an issue with `let` statements causing a SIGSEGV for the Bitwuzla solver. As part of that, I contributed to reporting that issue on their repository. [1] Bitwuzla is a relatively new, but performant and robust solver – therefore finding a significant bug of this form is considered quite unique and substantial. The issue has been solved in [2] by the respective maintainers, and incorporated into the main branch.