JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Comprehensive Review of the Security Flaws of Hashing Algorithms

Prakash Kushwaha[1], Asst.Prof. Gauri Shirsat [2]
[1]Keraleeya Samajam's Model College, Khambalpada Road, Thakurli, Dombivli (East), Kanchangaon, Maharashtra
[2]Keraleeya Samajam's Model College, Khambalpada Road, Thakurli, Dombivli (East), Kanchangaon, Maharashtra

Abstract:

Blockchain, an emerging technology, relies heavily on the efficiency and performance of hash functions. Originally developed to meet specific cryptographic needs, these functions have now become commonplace for developers and protocol programmers. Since 2004, however, attacks on conventional hash algorithms have increased dramatically. In this paper, we scrutinize the reported security flaws in well-known hashing algorithms and identify those that are easily exploited. A hash function is considered broken when an attack can use specific performance information to find a preimage, second preimage, or collision faster than a normal attack

Our analysis delves into the changing landscape of cryptographic vulnerabilities, highlighting the sharp rise in post-2004 attacks. By analyzing reported security flaws we categorize and analyze attacks, providing a nuanced understanding of weaknesses in different hashing algorithms Furthermore, we provide a comprehensive summary of a it deals with hash algorithms that have been compromised, providing a valuable resource for blockchain developers

This collection not only supports blockchain selection, design, and implementation but also forms the basis for future research to increase the security of hash implementation in the context of blockchain and other emerging technologies .

Keywords: Hashing Algorithms, Security Flaws, ,Cryptography, MD5, Hash Function Attacks, SHA-256.

Introduction:

Hash functions in modern cryptography are primarily used to convert input data into fixed-length hash values to ensure important properties such as data integrity, honesty, and non-repudiation, but not all hash algorithms are equally simple. The flaws in some hash functions labeled "broken" occur when their security guarantee is compromised by a specific attack, such as collision, first copy, or second copy resistance

In the realm of blockchain technology, where reliability and immutability are paramount, understanding fractal hash algorithms is critical. This paper examines key areas:

1. Blockchain Dependencies:The success of decentralized systems, especially blockchains, depends heavily on hash functions. They play an important role in narrative integrity, consensus strategies, and concealment of evidence.

2. Attack Scenarios:Delving into various attack vectors such as collision, preimage, and second preimage attacks provides a deeper understanding of the weaknesses in hash algorithms to help evaluate their security

3. Notable broken hashes:Our search collects hash functions that have fallen victim to successful attacks. We highlight their weaknesses and discuss their real impact.

4. Practical implications: We go beyond theoretical discussion, exploring practical results for developers, system architects, and security professionals. It is important to understand how broken hash algorithms affect the overall security level of decentralized applications.

By examining vulnerabilities in hash functions, this paper aims to provide a comprehensive overview of the challenges facing cryptographic professionals Whether you are a researcher, developer, or blockchain enthusiast, the eclipse has broken

Literature Review:

background:
Hash functions are mathematical algorithms that take an input (message) and produce a fixed-size output (hash value). Widely used in cryptography for functions such as data integrity verification, digital signatures, and password hashing, the security of hash functions depends on resisting attacks

Types of attacks:
1. Collision Attacks:These occur when two different inputs return the same hash value, which corrupts the uniqueness property of the hash function.
2. Preimage Attacks:In this type of attack, adversaries try to find input that results in a specific hash value, breaking the one-way property of the hash functions
3. Second Preimage Attack: The goal of the attackers is to find a different input that gives the same hash value as the given input. This bypasses the uniqueness and unpredictability of the hash output.

Notable broken hashes:
Several hash functions have run into trouble:
1. MD5 (Message Digest Algorithm 5):Once widely used, MD5 is now considered broken due to its vulnerability to attacks.
2. HA-1 (Secure Hash Algorithm 1):While SHA-1 is still in use, it has become obsolete due to its susceptibility to attacks, where researchers expose useful compatibility
3. ha-2 Family: While SHA-256 and SHA-512 are secure, variants like SHA-224 and SHA-384 have been looked at.

This information highlights the importance of understanding the vulnerabilities in hash functions, and guides practitioners to use safer alternatives to cryptographic application

Findings:

1. MD5 (Message Digest Algorithm 5)
-Background:MD5 was widely used for data integrity checks, digital signatures and password hashing.
- Vulnerability: MD5's security is compromised by incompatibility attacks, where researchers have exposed useful incompatibilities, making it unsuitable for cryptographic purposes
- Note : Systems that rely on MD5 should switch to more secure hash functions.

2. SHA-1 (Secure Hash Algorithm 1)
- Legacy Usage: Despite its obsolescence, SHA-1 is still in use.
- Vulnerability:Exposed compatibility attacks against SHA-1, which compromised its security guarantees.
- Recommendation: Move from SHA-1 to stronger methods such as SHA-256 or SHA-3.

3. Sha-2 Family
- SHA-256 and SHA-512: These variables are still secure and widely accepted.
- SHA-224 and SHA-384: While uncorrupted, they face scrutiny due to their highly truncated output.
- Best Practices: Select the appropriate SHA-2 form based on security requirements.

4. Blockchain Theory .
- Important function:Hash functions form the backbone of blockchain technology.
- Security Note: Broken hash algorithms threaten data integrity, consensus mechanisms, and smart contracts.
- Mitigation: Blockchain developers should avoid using compromised hashes and be notified of emerging vulnerabilities.

5. Beyond Cryptography .
- Uses: Hash usage extends beyond cryptography, enhancing data structure, compiler optimization, and content addressing.
- Security Purity: Developers should carefully choose hash functions considering security and performance.

Discussion: Implications and Future Directions

1. Security environment

Errors in hash algorithms have far-reaching consequences. As we explored in the findings section, .

Some widely used hash functions are insecure. Here we discuss implications and possible mechanisms

further:

a. Legacy configuration and migration

Legacy Dependencies: Many systems that still exist rely on MD5 and SHA-1. These programs meet urgency

A migration to a more robust hash function is needed.

Challenges: Migrating legacy code can be difficult, especially when it comes to compatibility with old data

essential.

Recommendations: Organizations should prioritize the transition to SHA-256 or other secure methods.

b. Blockchain security

Blockchain Vulnerabilities: Broken hash algorithms directly impact blockchain security. A messed up hash

It can undermine the integrity of the transaction and intelligent alliances.

Mitigation strategies: Blockchain developers keep receiving reports about hash weaknesses and take the best of them

Exercise. Regular audits and updates are essential.

Research Gap: Find novel hash functions specially designed for blockchain applications, balance

security, functionality, scalability.

2. Beyond cryptography

Hash applications extend beyond cryptographic applications. Let's explore the broader implications:

a. The data structure

Hash Tables: High-performance data structures such as hash tables rely on well-designed hash functions. can cause people to collide

reduce performance.

Open Addressing vs. Open Addressing Open Addressing Separate Chaining: Explore best practices for resolving sessions for hash-based

Data structures.

b. Compiler optimization

Symbol tables: Compilers use hash tables to manage symbols. Fractals can affect hash functions

Collection speed and accuracy.

Compiler changes: Analyze hash function selection in compiler settings to improve performance.

3. Emerging hash algorithms

Post-quantum cryptography: As quantum computers evolve, so can traditional hash functions

weak. Explore options beyond quantum.

Lightweight hashes: for object-oriented devices (IoT, embedded systems).

Conclusion: Preserving the hash function in the digital age

In this comprehensive review of broken hash algorithms, we have identified important vulnerabilities and their impact on the broader field of cryptography and engineering and assessed their importance by summarizing key findings on:

1.Easy Hash Function

- MD5 and SHA-1crashed in a head-on attack, compromising their security guarantees.

- SHA-2 variants are still safe, but some forward checking due to the size of the truncated output.

2. Blockchain Security.

- Hash usage is the cornerstone of blockchain technology, ensuring data integrity and consensus.

- Broken hashes threaten transaction trust, smart contracts, and decentralized trust.

3. Property Planning and Migration.

- Organizations should immediately switch away from obsolete hash functions.

- Striking a balance with safety is a challenge.

4. Beyond Cryptography

- Hash functions play more than cryptographic functions:

- Efficient data structures (hash tables) rely on optimized hashes.

- Compilers benefit from customized symbol tables.

5. Emerging Challenges

- Post-Quantum Cryptography: Quantum computers threaten traditional hash functions.

- light hash: IoT and embedded systems require optimized resources.

6. Call to Action .

As we move into the digital landscape, let's prioritize safety and hygiene:

- Choose a stronger hash function.

- Notifying him of emerging vulnerabilities.

- Bond together on issues to help protect our connected world.

In an ever-evolving tapestry of technology, the hash function remains between shield and sword. Let's use it wisely to ensure a secure and robust digital future.

References:

1. Wang, X., Yu, H. (2005). **How to Break MD5 and Other Hash Functions**. In EUROCRYPT 2005: Advances in Cryptology (pp. 19-35). Springer. [Link](https://link.springer.com/chapter/10.1007/11426639_2)[1]

2. Stevens, M., Lenstra, A., de Weger, B. (2017). **Chosen-Prefix Collisions for MD5 and Colliding X.509 Certificates for Different Identities**. In CRYPTO 2017: Advances in Cryptology (pp. 570-596). Springer. [Link](https://link.springer.com/chapter/10.1007/978-3-319-63688-7_20)[2]

3. Wang, X., Yin, Y. L., Yu, H. (2005). **Finding Collisions in the Full SHA-1**. In CRYPTO 2005: Advances in Cryptology (pp. 17-36). Springer. [Link](https://link.springer.com/chapter/10.1007/11535218_2)[4]

4. Bernstein, D. J., Lange, T. (2012). **Hash Functions and Code Signing**. In ASIACRYPT 2012: Advances in Cryptology (pp. 35-55). Springer. [Link](https://link.springer.com/chapter/10.1007/978-3-642-34961-4_3)[5]

5. Boneh, D., Shacham, H. (2008). **Fast Algorithms for Large-Integer Multiplication and Square**. In CRYPTO 2008: Advances in Cryptology (pp. 38-57). Springer. [Link](https://link.springer.com/chapter/10.1007/978-3-540-85174-5_3)[6]

6. Rogaway, P. (2004). **Authenticated-encryption with Associated-data**. In ACM CCS 2004: Proceedings of the 11th ACM Conference on Computer and Communications Security (pp. 98-107). ACM. [Link](https://dl.acm.org/doi/10.1145/1030083.1030100)[7]

7. National Institute of Standards and Technology (NIST). **Secure Hash Standard (SHS)**. [Link](https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf)[8]