KENNESAW STATE UNIVERSITY

COLLEGE OF COMPUTING AND SOFTWARE ENGINEERING

CS 4850-01 | FALL 2023 | SP-24 RED | SECURITY IN BLOCKCHAIN

UNDERLYING CRYPTOSYSTEM VULNERABILITIES

NIKKI DULANEY

INSTRUCTOR: SHARON PERRY

INTRODUCTION

A cryptosystem consists of a pair of algorithms that take a key to convert plaintext- readable messages that need protection- to ciphertext- messages that appear random and nonsensible- and then back to plaintext.

Underlying cryptosystem vulnerabilities are vulnerabilities that relate to a blockchain platform, as opposed to vulnerabilities associated with the programs deployed in a blockchain. These vulnerabilities are considered core blockchain vulnerabilities. Consensus mechanism manipulation and improper blockchain magic variation also fall into that category.

Underlying cryptosystem vulnerabilities can be found in components such as blockchain wallets, which work with public and private key pairs for signatures. Examples of underlying cryptosystem vulnerabilities include consensus algorithm vulnerabilities, digital signature weaknesses, key management issues, and hash function vulnerabilities.

METHODS OF DETECTION

These underlying cryptosystem vulnerabilities can be detected through security audits, implementation of monitoring and anomaly detection, vulnerability scanners, penetration testing, code reviews, fuzz testing, cryptanalysis, and red teaming.

Security Audits: Conducting security audits allows vulnerabilities to be identified and mitigated promptly.

Monitoring and Anomaly Detection: Implementing monitoring and anomaly detection systems can lead to finding unusual behavior or attacks.

Vulnerability Scanners: Using vulnerability scanners, which are automated tools, allows organizations to identify weaknesses in their networks, systems, and applications.

Penetration Testing: Authorizing simulated attacks on a computer system can allow exploitable vulnerabilities in a system to be detected.

Code Reviews: Examining source code, manually or through automation, can lead to the detection of vulnerabilities.

Fuzz Testing: Fuzzing injects invalid or unexpected inputs into a system to detect software vulnerabilities; it is an automated software testing method.

Cryptanalysis: Studying the cryptographic algorithm helps with understanding the cryptosystem and detecting vulnerabilities.

Red Teaming: Emulating real attack techniques and procedures against a system can result in the detection of vulnerabilities.

EXAMPLES

Consensus Algorithm Vulnerabilities

The consensus mechanism ensures agreement between all participants of the state of the ledger and the validity of transactions; popular consensus algorithms include Proof of Work (PoW) and Proof of Stake (PoS).

Vulnerabilities

Compromised consensus algorithms harm the integrity of the system and can lead to Sybil attacks, 51% attacks, timejacking, long-range attacks, and nothing-to-lose attacks.

Timejacking: Attackers can delay networks and disrupt consensus by manipulating the time stamp of their blocks.

Long-Range Attacks: Attackers can reconstruct a different blockchain from the genesis block, allowing them to take over it.

Nothing-to-Lose Attacks: Attackers without a stake in the network can create forks or withhold valid blocks to disrupt consensus.

Prevention Methods and Countermeasure Solutions

Preventative measures against these underlying cryptosystem vulnerabilities include network diversity, updating consensus algorithms, governance mechanisms, and incentive structures.

Network Diversity: A diverse range of participants on a decentralized network reduces the chance for collision attacks or Sybil attacks.

Updating Consensus Algorithms: Updating and improving consensus algorithms will help

prevent known vulnerabilities and enhance security overall.

Governance Mechanisms: Establish robust governance frameworks in order to make decisions and resolve disputes.

Incentive Structures: Economic incentives can cause participants' interests to align with the network's, thus discouraging desire to attack.

Digital Signature Weaknesses

The encryption process of digital signatures falls into two categories, asymmetric and symmetric encryption. The former uses different keys for encryption while the latter refers to systems in which both keys are similar. The asymmetric system is the most widespread; it can be broken down into two steps: signature and verification.

Vulnerabilities

Digital signatures introduce multiple security vulnerabilities. For instance, private keys can be stolen or impersonated. Additionally, cybercriminals can exploit weaknesses found in the software surrounding the creation and verification of digital signatures. Moreover, they might choose to manipulate the signed data between the creation and verification of the digital signature. The common vulnerabilities of digital signature weaknesses include the insecure storage of cryptographic keys and susceptibility to phishing attacks.

Insecure Storage: Examples include storing passwords in plaintext, using weak or reversible encryption algorithms such as MD5 or SHA-1, and using encryption keys that are too short or too weak.

Prevention Methods and Countermeasure Solutions

One prevention method is detecting fraud before it occurs. This can be done through the use of anomaly detection, forgery detection, and real-time monitoring.

Forgery Detection: Train machine learning algorithms to recognize differences between genuine and forged digital signatures, no matter how subtle.

Real-Time Monitoring: Monitor digital signature processes in real time in order to catch early warnings of security breaches or fraud.

Users can enable multi-factor authentication, use secure key storage, and enact robust

password policies. It is also important to keep software systems up to date, perform audits regularly, encrypt sensitive data, and stay informed on phishing attempts and attacks. Furthermore, it is useful to take advantage of trusted certificate authorities (CA) and certificate revocation lists (CRL). These actions are vital to prevention and countering.

Digital Certificate: An electronic ID card that provides authentication and encryption, thus it protects integrity.

CA: A trusted entity that validates the identities of entities, such as websites or email addresses, and binds them to key pairs with digital certificates.

CRL: A list of digital certificates that were revoked by the issuing certificate authority before they were assigned to expire. This makes it clear that a site's digital certificate is untrustworthy.

Key Management Issues

Key management involves generating, distributing, and protecting cryptographic keys. Key management issues are another example of underlying cryptosystem vulnerabilities. One compromised key could lead to a major data breach.

Vulnerabilities

Dangers and vulnerabilities to consider are weak keys, re-use of keys, incorrect use of keys, non-rotation of keys, non-destruction of keys, inadequate protection of keys and inappropriate storage of keys, insecure movement of keys, and insider threats.

Weak Keys: Keys need to be long enough for their intended purpose and generated with a high-quality random number generator (RNG). Implementing RNG poorly makes weak keys.

Incorrect Use: A key should have one specific purpose; if it is not used for its intended purpose, it might fail to provide the expected or required level of protection.

Non-rotation: A key that is overused becomes vulnerable to cracking; if compromised, a high volume of data could be exposed. This is caused by failing to rotate (update) keys at appropriate intervals.

Non-destruction: If keys are not destroyed- securely deleted without trace- after expiration, data can be compromised at a future date.

Inadequate Protection: Keys should be encrypted when stored if the value of the data calls for it

and should only be available in unencrypted form within an environment that is secure, and tamper protected. Otherwise, cryptographic keys could be exposed in server memory.

Insecure Movement: It can be necessary to move a key between systems. In these cases, the key should be encrypted under a pre-shared transport key- a key encryption key (KEK). The key should be split into separate, multiple components if that is not possible. Afterward the components are destroyed.

Insider Threats: A rogue employee with unrestrained access to a key can cause major vulnerabilities. They might also attempt to exploit vulnerabilities with limited access or by working with other rogue employees.

Prevention Methods and Countermeasure Solutions

The most effective way to prevent and counter key management issues is to utilize a dedicated electronic key management system, which should use a hardware security module to generate and protect keys and overall support the security of the system.

Hash Function Vulnerabilities

The final example of underlying cryptosystem vulnerabilities are hash function vulnerabilities. Hash functions are functions that convert a given key, numeric or alphanumeric, to a small integer value. It maps the number or string to a small integer that can be used as an index in a hash table.

Every given hash has preimages- or multiple inputs- that can produce it. A hash function creates new values according to a hashing algorithm.

Popular hash algorithms include MD5 (message-digest 5), RSA (Rivest, Shamir, Adleman), SHA (secure hash algorithm), and Ethash.

MD5: Used for message authentication, content verification, and digital signatures. However, it is considered an insecure algorithm.

RSA: Supports the integrity, confidentiality, non-reputability, and authenticity of data storage and digital communications. It is considered highly secure.

SHA: A family of cryptographic hash functions; a modified version of MD5 used for keeping data secure. Used to encrypt passwords and to detect tampering of data, for example.

Ethash: Algorithm used and performed by the Ethereum network.

Vulnerabilities

Vulnerabilities include collision attacks, preimage attacks, and second preimage attacks.

Collision Attacks: Occurs when the same hash is produced from different data inputs.

Preimage Attacks: Occurs when an attacker reverses the hash to find the original input.

Second Preimage Attacks: Occurs when an attacker uses one input to find a matching input that has the same hash output. This highlights the hash function's resistance to producing unique outputs is weak.

Prevention Methods and Countermeasure Solutions

Detection and prevention methods include security audits, cryptanalysis, testing for collisions, and benchmarking. Countermeasure solutions include using strong hash functions, updating hash function libraries and implementations regularly, strengthening keys through salting and multiple interactions, and monitoring vulnerability reports.

Collision Testing: Testing a hash function for collision resistance by searching for two different inputs that rest in the same hash is one way to spot the weakness early and prevent it.

Benchmarking: Assessing the speed and efficiency of hash functions can reveal potential vulnerabilities.

Strong Hash Functions: Using widely tested and validated hash functions like SHA-256 or SHA-3 can help prevent insecure hashes. Avoid outdated hash functions such as MD5 and SHA-1.

Salting: Adding random data, "salt" to a password before hashing ensures that if multiple users have the same password, their hash and salted passwords will not be the same.

Multiple Iterations: Applying a hash function to a password repeatedly creates a series of hash values in a chain. This increases the computational resources and time necessary for an attacker to crack a password. Also known as key stretching.

Monitoring Vulnerability Reports: Staying updated on hash function vulnerabilities through research and security advisories can allow prompt action in the event of vulnerability discovery.

CONCLUSION

In conclusion, underlying cryptosystem vulnerabilities are core blockchain vulnerabilities. The examples discussed in this paper were consensus algorithm vulnerabilities, digital signature weaknesses, key management issues, and hash function vulnerabilities. Methods of detection were identified, the vulnerabilities were expanded on, and prevention methods and countermeasure solutions were suggested. Identifying vulnerabilities is important as it allows potential threats and weaknesses to be addressed, which improves security and reliability. Furthermore, it allows the blockchain system to be built with stronger resistance to attacks, which aids in avoiding financial losses.

REFERENCES

Aakashyap. "Insecure Cryptographic Storage." *Medium*, Medium, 24 Apr. 2023, medium.com/@aakashyap_42928/insecure-cryptographic-storage-fe5d40d10765#:~:text=Examples%20of%20insecure%20cryptographic%20storage,of%20a%2025 6-bit%20key.

Anderson, Evan. "Red Teaming 101: What Is Red Teaming?" *IBM Blog*, 4 Oct. 2023, www.ibm.com/blog/red-teaming-101-what-is-red-teaming/.

Awati, Rahul, and Michael Cobb. "What Is a Certificate Revocation List (CRL) and How Is It Used?" *Security*, TechTarget, 20 Aug. 2021, www.techtarget.com/searchsecurity/definition/Certificate-Revocation-List#:~:text=A%20certificate%20revocation%20list%20(CRL)%20is%20a%20list%20of%20digital,actual%20or%20assigned%20expiration%20date.

Bellovin, Steven. *What Is a Cryptosystem? - Department of Computer Science, Columbia ...*, www.cs.columbia.edu/~smb/classes/f06/l03.pdf. Accessed 17 Oct. 2023.

Bob. "Preventing Digital Signature Fraud: Tips and Strategies." *Imagine IT*, 7 Aug. 2023, imagineiti.com/preventing-digital-signature-fraud/.

Christophe. "Key Stretching Concepts and Algorithms - SY0-601 Comptia Security+." *Cybr*, 1 Sept. 2023, cybr.com/certifications-archives/key-stretching-concepts-and-algorithms/.

"Cryptanalysis and Types of Attacks." *GeeksForGeeks*, www.geeksforgeeks.org/cryptanalysis-and-types-of-attacks/amp/. Accessed 17 Oct. 2023.

CybelAngel. "Digital Signatures Are the Cybersecurity Vulnerability You Need to Stop Ignoring." *CybelAngel*, 18 July 2023, cybelangel.com/digital-signatures-are-the-cybersecurity-vulnerability-you-need-to-stop-ignoring/.

Devi, Navanita. "What Is a Salt and How It Boosts Security?" *Loginradius*, www.loginradius.com/blog/identity/what-is-salt/. Accessed 17 Oct. 2023.

"Hash Functions and List/Types of Hash Functions." *GeeksForGeeks*, www.geeksforgeeks.org/hash-functions-and-list-types-of-hash-functions/amp/. Accessed 17 Oct. 2023.

Lake, Josh. "What Is a Collision Attack?" *Comparitech*, 13 Sept. 2023,

www.comparitech.com/blog/information-security/what-is-a-collision-attack/.

Lake, Josh. "What Is a Preimage Attack and Are They Dangerous?" *Comparitech*, 14 June 2022, www.comparitech.com/blog/information-security/what-is-preimage-attack/#:~:text=If%20a%20cryptographic%20hash%20function,original%20input%20from%20a%20hash.

Nagaraj, Karthikeyan. "Consensus Vulnerability in Blockchain: Understanding, Exploitation, and Prevention: 2023." *Medium*, Medium, 26 May 2023, cyberw1ng.medium.com/consensus-vulnerability-in-blockchain-understanding-exploitation-and-prevention-2023-46d53b6cb946.

Sookman, Barry B. "Blockchain Vulnerabilities - Crypto Hacks, Blockchain Forensics and Legal Challenges." *Lexology*, McCarthy Tétrault LLP, 19 Nov. 2021, www.lexology.com/library/detail.aspx?g=d149175e-e73b-4b49-855a-54df7ddbb34c.

Streichsbier, Stefan. "Insecure Hash." *GuardRails*, Feb. 2023, docs.guardrails.io/docs/vulnerability-classes/insecure-use-of-crypto/insecure-hash#:~:text=prevent%20insecure%20hashes%3F-,%E2%80%8B,-256%20or%20SHA-3.

Stubbs, Rob. "Cryptographic Key Management - the Risks and Mitigation." *Cryptomathic*, Cryptomathic, 23 Mar. 2023, www.cryptomathic.com/news-events/blog/cryptographic-key-management-the-risks-and-mitigations.

Team, SSL.com Support. "What Is a Certificate Authority (CA)?" *SSL.Com*, 9 Dec. 2021, www.ssl.com/faqs/what-is-a-certificate-authority/amp/.

Vidal, Fernando Richter, et al. "OpenSCV: An Open Hierarchical Taxonomy for Smart Contract Vulnerabilities." *arXiv.Org*, 7 Apr. 2023, arxiv.org/abs/2303.14523.

"What Are the Popular Hashing Algorithm?" *Online Tutorials, Courses, and eBooks Library*, 2022, www.tutorialspoint.com/what-are-the-popular-hashing-algorithm.

"What Are Vulnerability Scanners and How Do They Work?" *CSO Online*, 10 Apr. 2020, www.csoonline.com/article/569221/what-are-vulnerability-scanners-and-how-do-they-work.html/amp/.

"What Is Fuzz Testing and How Does It Work?" *Synopsys*, www.synopsys.com/glossary/what-is-fuzz-testing.html#:~:text=Definition,as%20crashes%20or%20information%20leakage. Accessed 17

Oct. 2023.

"What Is Penetration Testing and How Does It Work?" *Synopsys*, www.synopsys.com/glossary/what-is-penetration-testing.html. Accessed 17 Oct. 2023.

"What Is Secure Code Review and How Does It Work?" *Synopsys*, www.synopsys.com/glossary/what-is-code-review.html#:~:text=Secure%20code%20review%20is%20a,style%20guidelines%2C%20among%20other%20activities. Accessed 17 Oct. 2023.