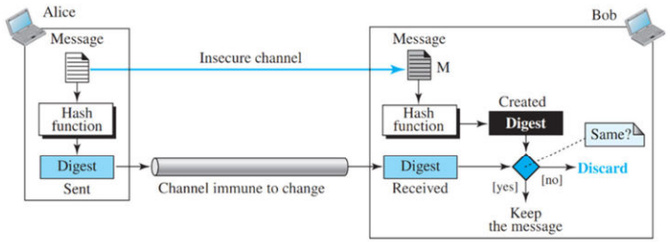
Name: _____ **Student No.:** _____ **Date:** __/__/____

Cryptographic Hash Functions

CRYPTOGRAPHIC HASH FUNCTIONS (CHFs)

A cryptographic hash function **converts any input** (such as a message) into a **fixed-length, n-bit string** called a **digest**.



It allows you to **verify data without exposing the original content.**

In this worksheet, you will:

1. Explore **various cryptographic hash functions** (CHFs) and their **uses.**
2. Understand why **MD5** and **SHA-1** are **outdated**.
3. Learn how hash functions help with **password storage** and **file verification.**

What are the most commonly used CHFs?

The following table lists some commonly used cryptographic hash functions, along with their hash lengths and primary application domains:

Hash Function	Output Length (bits)	Common Applications
MD5	128	File integrity checking, older systems, non-crypto uses
SHA-1	160	Legacy systems, Git for version control
SHA-256	256	Cryptocurrency (Bitcoin), digital signatures, certificates
SHA-3	Variable (up to 512)	Various cryptographic applications, successor to SHA-2
Blake2	Variable (up to 512)	Cryptography, replacing MD5/SHA-1 in some systems
Blake3	Variable (up to 256)	Cryptography, file hashing, data integrity

Notes:

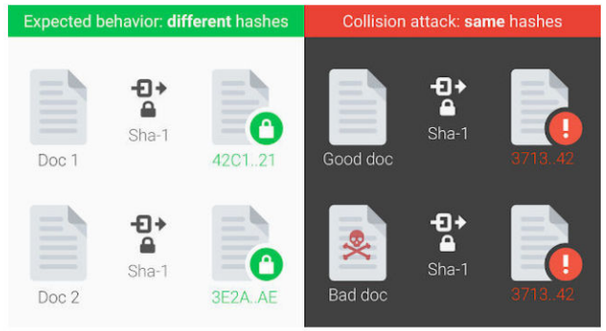
- **MD5 and SHA-1** are **deprecated** because they're **vulnerable to collisions** and **shouldn't be used in new systems.**
  - A **good CHF** should **resist collisions**, making it **hard to find two inputs with the same hash.**
- **SHA-3**, chosen by NIST in 2015, is a **secure alternative to SHA-2** with **different internal characteristics** and **added resistance to certain attacks.**
- **Blake2 and Blake3** are **faster than MD5, SHA-1, SHA-2, and SHA-3**, while **matching SHA-3's security.**
  - They're **increasingly popular in new systems where speed is crucial.**

Let us verify that the Blake algorithm is faster than the other specified CHFs:

1. Scan the **QR code** to open the **Google Colab notebook.**
2. **Duplicate the notebook for editing** by selecting **File > Save a Copy in Drive.**
3. In your copy, **enter your name and section** by editing the designated text cell.
4. **Follow the instructions and answer all questions thoroughly.**

Are MD5 and SHA-1 still secure?

According to OWASP, it's best to **avoid deprecated cryptographic functions** like MD5 and SHA-1 because they are **vulnerable to collision attacks** -where **two different inputs produce the same hash.**



This vulnerability poses a critical security risk. For example, the **Flame malware** exploited this weakness by **using a forged Microsoft certificate. Since Windows**

Update verified updates using MD5-based signatures, the malware appeared as a legitimate update, tricking systems into installing it.

Let us demonstrate the vulnerability of SHA-1 to collision attacks through file verification.

1. Scan the **QR code** to open the **Google Colab notebook.**
2. **Duplicate the notebook for editing** by selecting **File > Save a Copy in Drive.**
3. In your copy, **enter your name and section** by editing the designated text cell.
4. **Follow the instructions and answer all questions thoroughly.**

What are some common uses for CHFs?

CHFs are used not just for file verification but also for **secure password storage.**

A basic approach to authentication is **storing usernames and passwords in a database.** When a user logs in, the server checks if the provided credentials match the stored ones.

However, **storing passwords in plaintext is risky—if hackers access the database, they can steal every password.** A safer method is **hashing**, which converts passwords into irreversible data, making them unreadable even if stolen.

bcrypt

There are many cryptographic functions, like SHA-2 and SHA-3, but they were **designed for speed. The faster the function, the faster an attacker can brute-force passwords**- modern hardware can compute millions or even billions of SHA-256 hashes per second.

For password security, we need a slow, adaptive hashing function that can be adjusted over time as hardware improves. bcrypt was designed for this purpose. Its slow hashing process limits the number of guesses an attacker can make per second, making brute-force attacks much harder.

bcrypt also enforces best practices by requiring a salt, which protects against rainbow table attacks.

Resources:

Cryptography and Network Security, Stallings

Data Communications and Networking, Forouzan

https://owasp.org/Top10/A02_2021-Cryptographic_Failures/

https://auth0.com/blog/hashing-in-action-understanding-bcrypt/

https://auth0.com/blog/hashing-passwords-one-way-road-to-security/

https://learning.quantum.ibm.com/course/practical-introduction-to-quantum-safe-cryptography/cryptographic-hash-functions

https://blog.cloudflare.com/why-its-harder-to-forge-a-sha-1-certificate-than-it-is-to-find-a-sha-1-collision/