Blake2 Algorithm

Introduction

Blake2 is a successor to the Blake hash function and is designed to be faster and more secure than its predecessor. It comes in two variants: Blake2b and Blake2s.

Blake2b is optimized for 64-bit platforms and produces a 512-bit hash value. It has a block size of 128 bytes and uses a tree-based hashing mechanism that allows it to hash large amounts of data quickly. It also supports keying and personalization, which allows the user to generate different hash values for the same input data by using different keys or personalization parameters.

Blake2s, on the other hand, is optimized for 32-bit platforms and produces a 256-bit hash value. It has a block size of 64 bytes and uses a similar tree-based hashing mechanism as Blake2b. Like Blake2b, it also supports keying and personalization.

Both variants of Blake2 are designed to be secure against a wide range of attacks, including collision attacks, preimage attacks, and second-preimage attacks. They also have built-in support for parallel processing, which allows them to take advantage of modern multicore processors to further increase their speed.

Blake2 has become a popular choice for many cryptographic applications, including password storage, message authentication, and digital signatures. It is widely used in many popular software libraries, including OpenSSL, libsodium, and Golang's standard library.

Literature Survey

BLAKE is a cryptographic hash function based on Daniel J.
Bernstein's ChaCha stream cipher, but a permuted copy of the input block, XORed with round constants, is added before each ChaCha round.
Like SHA-2, there are two variants differing in the word size. ChaCha operates on a 4×4 array of words. BLAKE repeatedly combines an 8-word hash value with 16 message words, truncating the ChaCha result to obtain the next hash value. BLAKE-256 and BLAKE-224 use 32-bit words and produce digest sizes of 256 bits and 224 bits, respectively, while BLAKE-512 and BLAKE-384 use 64-bit words and produce digest sizes of 512 bits and 384 bits, respectively.

The BLAKE2 hash function, based on BLAKE, was announced in 2012.
The BLAKE3 hash function, based on BLAKE2, was announced in 2020.

Working of Blake2:-

The working of Blake2 is similar to Blake in that it divides the input message into fixed-size blocks and processes each block through a compression function that produces a fixed-size output. However, Blake2 uses a different compression function that is faster and more secure than the one used in Blake.

The Blake2 compression function consists of several rounds of mixing and permutation, which are similar to the ones used in Blake. However, it also includes a new feature called the message schedule, which allows it to take advantage of parallel processing and increase its speed.

The message schedule is a tree-like structure that allows the compression function to process multiple message blocks in parallel. Each leaf node of the tree corresponds to a single message block, and each internal node corresponds to the output of a previous round of mixing and permutation.

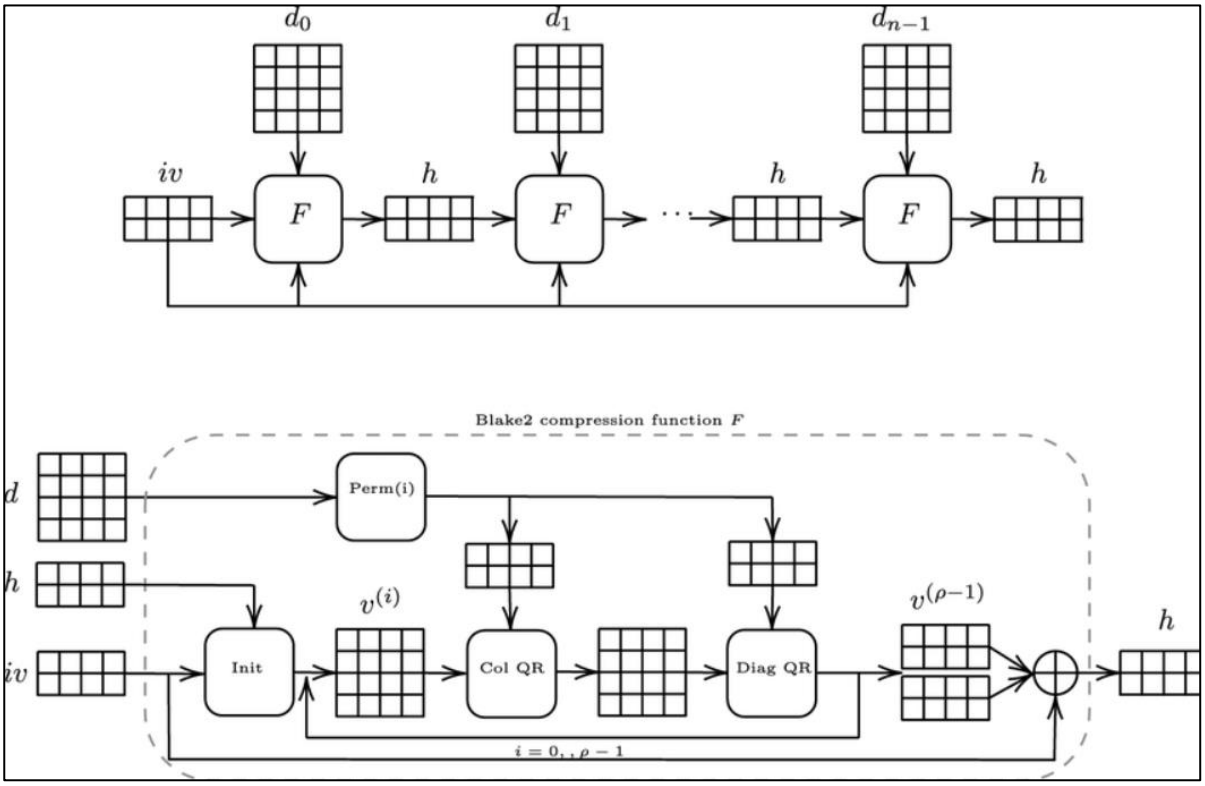
When the compression function is applied to a set of message blocks, it first builds the message schedule by hashing each block and combining the resulting hash values into a binary tree. It then applies the mixing and permutation rounds to the nodes of the tree in a top-down manner, starting from the root node and working its way down to the leaf nodes.

After all the rounds have been completed, the output of the compression function is the hash value for the set of input message blocks. This process can be repeated for each set of input message blocks to produce the final hash value.

Like Blake, Blake2 is designed to be resistant to a wide range of attacks, including collision attacks, preimage attacks, and second-preimage attacks. It also supports keying and personalization, which allows the user to generate different hash values for the same input data by using different keys or personalization parameters.

Specifications

- blake2.pdf is the original BLAKE2 documentation, which describes how blake went from the SHA-3 finalist BLAKE to BLAKE2, how all the BLAKE2 versions work, and analyses BLAKE2's performance and security.
- RFC 7693 is an RFC edited by Markku-Juhani O. Saarinen that provides a complete specification of BLAKE2b and BLAKE2s
- blake2x.pdf, the specification of BLAKE2X, versions of BLAKE2 to create hashes of any length up to 4 GiB and build XOFs, KDFs, and DRBGs (published to request comments, design not final yet)



Applications

Blake2 has a wide range of applications in various areas of cryptography and security. Some of the most common applications of Blake2 are:

1. Password hashing: Blake2 is commonly used for password hashing and storage due to its security features and speed. Passwords are hashed with a unique salt and stored in a secure database to prevent unauthorized access.
2. Message authentication: Blake2 is also used for message authentication, which involves verifying the integrity of a message to ensure that it has not been tampered with. This is useful in secure communication protocols and digital signature applications.
3. File integrity checking: Blake2 can be used to ensure the integrity of files and data by hashing the data and comparing it to a known hash value. This can be useful in detecting data tampering or corruption.
4. Key derivation: Blake2 can be used for key derivation, which involves generating a secure key from a source of randomness or a password. This is useful in encryption and decryption applications.
5. Blockchain and cryptocurrency: Blake2 is used in some cryptocurrencies and blockchain protocols, such as Siacoin and Decred, for generating secure and efficient hash functions.

6. Network security: Blake2 can also be used in network security applications, such as secure network protocols and secure web browsing, to provide secure and efficient hashing functions.

Other uses are as follows:-

- Linux kernel RNG: The Linux kernel's RNG uses BLAKE2s as its entropy extractor
- OpenSSL: OpenSSL includes BLAKE2b and BLAKE2s
- WireGuard: The WireGuard VPN uses BLAKE2s for hashing and as a MAC
- WolfSSL: WolfSSL includes BLAKE2b
- Botan: The Botan library includes BLAKE2b
- Crypto++: The Crypto++ library includes BLAKE2s and BLAKE2b
- Noise: The Noise protocol (used in WhatsApp and WireGuard) uses BLAKE2s and BLAKE2b
- Cifra Extrema: Cifra Extrema products use several versions of BLAKE2 in its servers and satellite modules

Password hashing schemes:

- Argon2 (by Biryukov, Dinu, Khovratovich; PHC winner)
- Catena (by Forler, Lucks, Wenzel; PHC candidate)
- Lanarea (by Mubarak; PHC candidate)
- Lyra and Lyra2 (by Simplicio Jr., Barreto, Almeida, Andrade; PHC candidate)
- Neoscrypt (by Doering)
- RIG (by Chang, Jati, Mishra, Sanadhya; PHC candidate)
- TwoCats (by Cox; PHC candidate)
- Yarn (by Capun; PHC candidate)

References

1. https://www.blake2.net/
2. https://en.wikipedia.org/wiki/BLAKE_(hash_function)
3. https://www.npmjs.com/package/blake2
4. www.youtube.com
5. https://chat.openai.com