**Argon2**

1. **Definition** : Argon2 is a key derivation function (KDF) and password hashing algorithm that was selected as the winner of the Password Hashing Competition (PHC) in 2015. It is designed to securely hash passwords and other sensitive information, making it difficult for attackers to reverse-engineer the original input from the hash value. Argon2 is known for its resistance against various cryptographic attacks, including brute-force and time-memory trade-off attacks. It is considered to be highly secure and is widely used for securely storing passwords and protecting sensitive data in software applications.
2. **How argon2 work** ?

1. \*\*Initialization:\*\* Argon2 takes several parameters as input, salt and so on . These parameters influence how much memory and time the algorithm will consume.

2. \*\*Memory-Hardness:\*\* Argon2 uses a large, configurable amount of memory.

3. \*\*Data-Dependent Operations:\*\* Argon2 operates on the input data (password and salt) , meaning the operations performed depend on the actual data being processed.

4. \*\*Parallel Processing:\*\* Argon2 can be parallelized efficiently. The parallelism factor parameter controls how many threads can be used to hash the password.

5. \*\*Iterations:\*\* Argon2 applies a specified number of iterations to the data. This makes the hash calculation intentionally slow, which is a desirable feature for password hashing algorithms.

6. \*\*Finalization:\*\* After the specified number of iterations, Argon2 produces the final hash value. This hash value can be stored in a database for later authentication purposes.

1. All three modes allow specification by three parameters that control:

* execution time
* memory required
* degree of parallelism

1. using with :

-PHP

- Nodejs

- Js

1. Main parameter to argon2
2. **Password:** The input password to be hashed.
3. **Salt:** A random value used to increase the security of the hash.
4. **Memory Cost:** The amount of memory (in kilobytes) that the algorithm uses.
5. **Time Cost:** The number of iterations the algorithm performs.
6. **Parallelism Factor:** The number of parallel threads or lanes used for computation.