



SeSAC 4기, 

웹 풀스택 과정 비밀번호 암호화 수업

WITH 팀 리처드



암호화



“Hello, CODINGOn!”

이메일

비밀번호

영문, 숫자, 특수문자 조합 8글자 이상

성함

동의하고 가입하기

서비스 이용약관 및 개인정보 처리방침에 동의합니다.

☒ 마케팅 정보 활용 동의(선택) 보기

간편 회원가입



이미 회원이신가요? 로그인하기



“Hello, CODINGOn!”

이메일

비밀번호

로그인

간편 로그인



아직 회원이 아니신가요? 회원가입하기

이메일을 잊으셨나요? 이메일찾기

비밀번호를 잊으셨나요? 비밀번호찾기

현재 DB

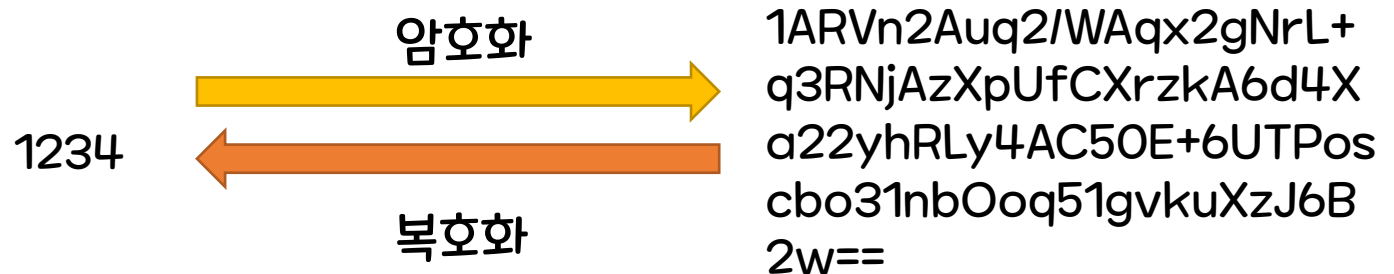
	id	pw	name
▶	1		1
	2		2
	dvadva	ks3ah	솔하나
	hanjo	jk4	한조
	hong1234	8o4	홍길동
	jungkrat	4ift	정크랫
	power70	qx38s	변사또
	sexysung	8awjko	성춘향
	widowmaker	3ewifh3	위도우
●	NULL	NULL	NULL

암호화 종류

단방향 암호화



양방향 암호화



암호화 종류

암호화 종류			
	암호화	복호화	암호화 방식
단방향	가능	불가능	Hash
양방향	가능	가능	대칭키(비공개키) 비대칭키(공개키)

해시(Hash)

- 해시(Hash) : 해시 함수에 의해 얻어지는 값
- 해시 함수 (Hash Function) = 해시 알고리즘
 - 키(key) : 매핑 전 원래 데이터 값
 - 해시 값 (hash value) : 매핑 후 데이터 값
 - 해싱 (hashing) : 매핑하는 과정



Bcrypt

- 비밀번호를 암호화하는 알고리즘 중 하나
- Blowfish 암호를 기반으로 설계된 암호화 함수
- 현재까지도 사용 중인 가장 강력한 매커니즘임과 동시에 해싱이 느리고 비용이 많이 든다.
- 강력한 보안이 필요할 때 적합

Crypto

- 암호화 알고리즘이 모여 있는 패키지

```
npm i crypto
```



```
const crypto = require("crypto");
```

Crypto

- 안전하지 않은 함수 : MD5, SHA-1, HAS-180
- 안전한 함수 : SHA-256 **SHA-512** 등



SHA-2 알고리즘의 512bit 버전

국가안보국(NSA)가 설계한 암호 해시함수로 512비트(64바이트) 해시 값을 생성하는데, 길이는 128자리 16진수로 렌더링된다.

Crypto 암호화

```
const crypto = require('crypto');

const createHashedPassword = (password) => {
  return crypto.createHash("sha512").update(password).digest("base64");
};
```

Crypto 암호화

```
const crypto = require('crypto');

const createHashedPassword = (password) => {
  return crypto.createHash("sha512").update(password).digest("base64");
};
```



해시를 만들기 위해 사용하는 함수로 parameter로는 사용할 알고리즘 이름이 들어간다.

Crypto 암호화

```
const crypto = require('crypto');

const createHashedPassword = (password) => {
  return crypto.createHash("sha512").update(password).digest("base64");
};
```



해시를 만들기 위해 사용하는 함수로 parameter로는 사용할 알고리즘 이름이 들어간다.

Crypto 암호화

```
const crypto = require('crypto');

const createHashedPassword = (password) => {
  return crypto.createHash("sha512").update(password).digest("base64");
};
```

해시를 만들 때 인코딩 방식을 설정한다.

base64 : 64진법

문자 코드에 영향을 받지 않는 공통 ASCII 영역의
문자들로만 이루어진 일련의 문자열로 바꿔주는 인코딩 방식

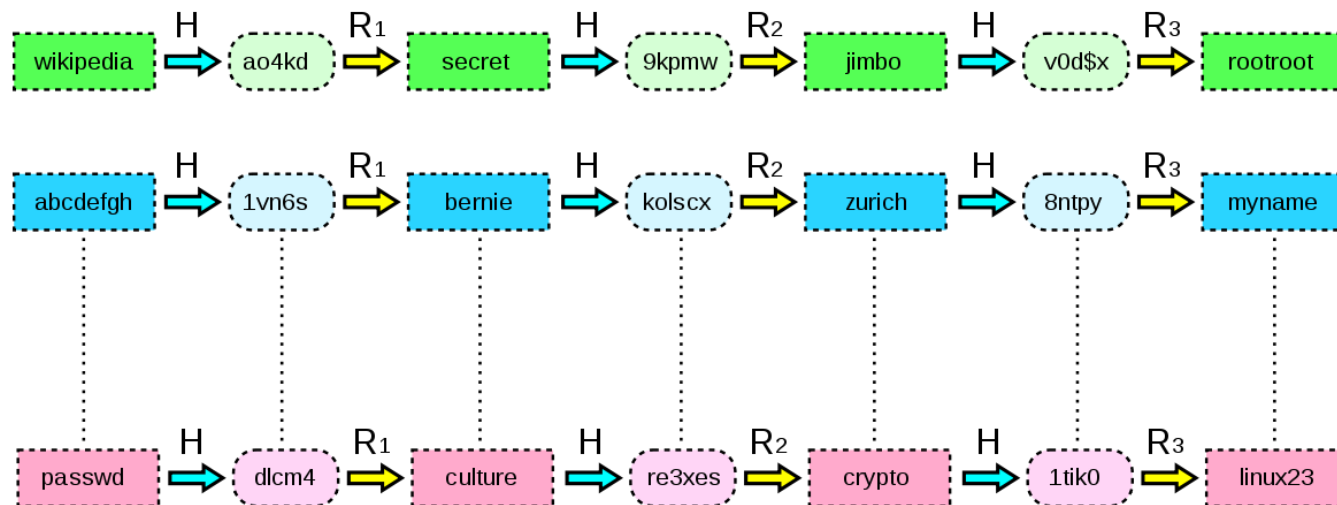
hex, latin1

레인보우 테이블

해시함수를 사용해 만들어낼 수 있는

값들을 대량으로 저장해놓은 표

즉, 보통 해시함수를 이용해 저장된 비밀번호로부터 원래의 비밀번호를 추출해내는데 사용된다.



암호화 보완법

1) salt

- 입력한 값에 salt라는 특정 값을 붙여 변형시키는 것

2) 해시 함수 반복

- 해시 함수를 여러 번 돌려 본래의 값을 예측하기 어렵게 만드는 것

Crypto 암호화

```
const createdHash = (password) => {  
  const salt = crypto.randomBytes(64).toString('base64');  
  return crypto.pbkdf2Sync(password, salt, 10, 64, 'sha512').toString('base64');  
}
```

Crypto 암호화

```
const createdHash = (password) => {  
  const salt = crypto.randomBytes(64).toString('base64');  
  return crypto.pbkdf2Sync(password, salt, 10, 64, 'sha512').toString('base64');  
}
```

crypto 에 내장되어 있는 함수로 parameter로 받은 바이트 길이로 salt를 생성한다.

Crypto 암호화

```
const createdHash = (password) => {  
  const salt = crypto.randomBytes(64).toString('base64');  
  return crypto.pbkdf2Sync(password, salt, 10, 64, 'sha512').toString('base64');  
}
```



해싱할 값(키), salt, 해시 함수 반복 횟수, 해시 값 길이, 해시 알고리즘

Crypto 검증

```
const verifyPassword = (password, salt, userPassword) => {  
  const hashed = crypto.pbkdf2Sync(password, salt, 10, 64, 'sha512').toString('base64');  
  
  if ( hashed === userPassword ) return true;  
  return false;  
}
```

Crypto 는 단방향 알고리즘이기 때문에 복호화가 불가능

즉, 입력한 값과 동일한 알고리즘을 이용해 다시 암호화를 해 비교한다.