보안

1. 쿠키(Cookie)

1.1 쿠키란?

- 웹 서버가 브라우저에게 지시하여 사용자의 로컬 컴퓨터에 파일 또는 메모리에 저장하는 작은 기록 정보 파일입니다.
- 파일에 담긴 정보는 인터넷 사용자가 같은 웹사이트를 방문할 때마다 읽히고 수시로 새로운 정보로 바뀔 수 있습니다.

1.2 쿠키의 구성요소

- Name 이름
- Value 데이터
- Expires 삭제기한
- Domain 쿠키가 사용되는 곳의 도메인
- Path 쿠키를 반환하는 경로
- Secure 보안 연결 등

1.3 쿠키의 단점

- 쿠키가 유출되어 보안 문제가 발생할 수 있습니다.
- 트래픽이 과중될 수 있습니다.

2. 세션(Session)

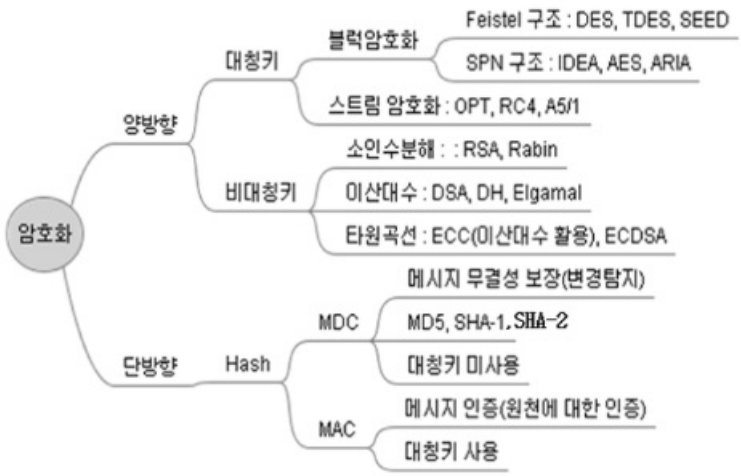
2.1 세션이란?

- 쿠키의 트래픽 문제와 쿠키를 변경하는 보안적 이슈를 해결하기 위해 등장한 것으로, HTTP Session id를 식별자로 구별하여 데이터를 사용자의 브라우저에 쿠키형태가 아닌 접속한 서버 DB에 정보를 저장하고 브라우저가 종료되면 사라집니다.

2.2 세션의 단점

- 세션 저장 장치가 부족한 시스템에는 적합하지 않습니다.
- load-balancing 문제가 발생합니다.

3. 대칭키 암호화(Symmetric-Key Cryptography)



3.1 대칭키 암호화란?

- 대칭키 암호화란 암호화에 사용되는 암호화키와 복호화에 사용되는 암호화 키를 같은 것으로 사용하는 암호화 기법입니다.

3.2 대칭키 암호화의 단점

- 대칭키를 상대방에게 알려주기 어렵습니다.
- 대칭키 유출 시 보안에 취약해집니다.

3.3 대칭키 암호화의 종류

- DES, AES 등이 있습니다.

4. 공개키 암호화(Public-Key Cryptography)

4.1 공개키암호화란?

- 공개키 암호는 한 쌍의 키가 존재하며, 하나는 특정 사람만이 가지는 개인키(또는 비밀키)이고 다른 하나는 누구나 가질 수 있는 공개키입니다. 개인키로 암호화 한 정보는 그 쌍이 되는 공개키로만 복호화 가능하고, 반대로 공개키로 암호화 한 정보는 그 쌍이 되는 개인키로만 복호화가 가능합니다. 즉 공개키 암호 방식은 암호화할 때 사용하는 암호키와 복호화할 때 사용하는 암호키가 서로 다르기 때문에 비대칭키 암호라고도 합니다.

4.2 공개키 암호화의 단점

- 대칭키에 비해 속도가 느립니다.
- 공개키와 개인키의 신뢰성을 보장하기 위해 인증기관으로부터 발급받아야합니다.

4.3 공개키 암호화의 종류

- RSA

5. 단방향 암호화(One-Way Cryptography)

5.1 단방향 암호화란?

- 단방향 암호화는 주로 해시함수를 이용하여 원래의 값을 복호화하지 없게 한 암호화 방법입니다. 이는 원래의 data 값을 확인할 수 없고 그저 암호화된 문자열로 사용됩니다. 주로 비밀번호의 원래 데이터값을 알 수 없게 하면서 해쉬함수를 통해 나온 값을 비교하여 인증하는데 쓰입니다.

6. OAuth 2.0

6.1 OAuth이란?

- oAuth는 서비스간의 상호작용을 용이하게 하기 위해서 사용하는 것으로 AccessToken(ID나 비밀번호가 아닌 토큰)을 주는 방식입니다.

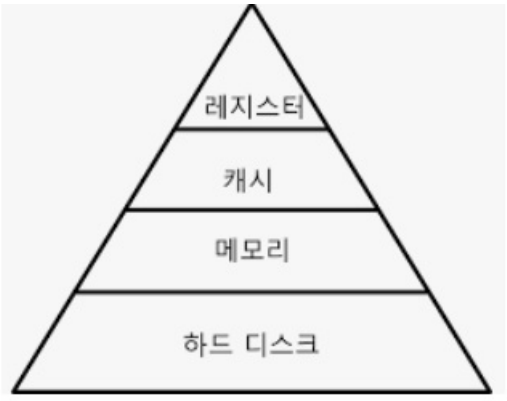
6.2 OAuth 인증과정

- 액세스토큰을 발급하기위해 가장 먼저 resourceOwner로부터 허락을 받습니다.
- resource Server로부터 인증을 받은 후 해당 Client ID가 있는지 확인합니다.
- 상자간의 통신이기 때문에 Resource Owner가 Client로 이동합니다.
- Client가 받은 authorization code를 가지고 Resource Server로 이동하여 인증절차를 거칩니다.
- Resource Server로부터 받은 accessToken를 사용하기 시작합니다.

컴퓨터 구조

1. 메모리 계층 구조(Memory Hierarchy)

1.1 메모리 계층 구조란?



- 메모리 계층 구조란 메모리를 필요에 따라 여러 종류로 나누어 놓은 것을 의미합니다.
- 접근 속도가 빠른 순으로 레지스터, 캐시 메모리, 메인메모리, 보조 기억장치로 나뉘어집니다.

1.2 메모리 계층 구조가 필요한 이유

- 프로그램이 동작할 때 쓰이는 자원을 효율적으로 사용하기 위해 메모리 계층 구조가 도입되었습니다.

1.3 레지스터

- 레지스터는 CPU 내부에 존재하며 연산을 수행할 때 참조되는 기억장치입니다.

1.4 캐시 메모리

- 캐시 메모리 또한 CPU 내부에 존재하며 IO버스를 통해 정보를 주고받습니다.
- 캐시 메모리는 L1 Cache, L2 Cache 등으로 분할하여 사용합니다.

1.5 보조기억장치

- 하드디스크나 USB 메모리 등을 말하며, 접근속도가 굉장히 느립니다.

2. 캐시(Cache)

2.1 캐시란?

- 프로그램이 수행될 때 나타나는 지역성을 이용하여 메모리나 디스크에서 사용되었던 내용을 특별히 빠르게 접근할 수 있는 곳에 보관하고 관리함으로써 이 내용을 다시 필요로할 때 보다 빠르게 참조하도록 하는 것입니다.
- CPU와 메인 메모리의 사이에서 속도 완충 역할을 합니다.(sram)

2.2 지역성

- 공간적 지역성 : 메인메모리에서 CPU가 요청한 주소지점의 데이터에 인접한 주소의 데이터들이 앞으로 참조될 가능성이 높음
- 시간적 지역성 : 한번 참조되었던 데이터는 후에 다시 참조될 가능성이 높음
- 순차적 지역성 : 분기(branch)가 없는 한 데이터가 기억장치에 저장된 순서대로 순차적으로 인출되고 실행될 가능성이 높음

2.3 Cache hit

- CPU가 데이터를 원할때 캐쉬 메모리 내에 해당 데이터가 존재할 때 발생하는 것으로 캐쉬 적중이라고 합니다.

데이터베이스

1. 관계형 데이터베이스(Relational Database)모델

1.1 관계형 데이터베이스란?

- 관계형 데이터베이스는 키와 값들의 관계를 테이블화 시킨 매우 간단한 원칙의 전산정보 데이터베이스이다.

1.2 관계형 데이터베이스 모델이란?

- 관계형 데이터베이스의 모델은 데이터를 컬럼과 로우를 이루는 하나 이상의 테이블로 정리하며, 고유 키로 각 로우를 식별한다.
- 로우는 레코드나 튜플로 부른다. 로우는 그 엔티티 종류의 인스턴스를 대표하며 컬럼은 그 인스턴스의 속성이 되는 값들을 대표한다.

1.3 관계형 모델의 제약사항

- 릴레이션(Relation)에서 각 튜플(Tuple)은 유일해야 한다.(원자성)
- 릴레이션 내의 튜플의 순서는 중요하지 않다.
- 모든 릴레이션은 함수 종속 규칙을 따라야 한다.
- 릴레이션이 정규화(Normalization)되지 않으면 관계형 모델이라 할 수 없다.
- 정규화된 릴레이션 사이에는 외래 식별자(Foreign Identifier)를 통해서 연관 관계가 성립하게 된다.

2. 데이터(Data)와 정보(Information)의 차이

2.1 데이터란?

- 현실 세계에서 벌어지는 일들을 단순한 방법으로 관찰하고 측정해서 얻은 값이다.
- 개별 데이터 자체로는 의미가 중요하지 않은 객관적 사실이다.

2.2 정보란?

- 그 데이터를 가공해서 얻을 수 있는 의사 결정에 도움을 줄 수 있는 유용한 형태의 결과값이다.
- 데이터 베이스를 이용하는 목적으로, 자료를 통해 정보로 만듭니다.

3. 테이블(Table), 튜플(Tuple) / 레코드(Record), 속성(Attribute) / 필드(Field), 뷰(View), 키(Key), 인덱스(Index)

3.1 테이블

- 같은 성격의 데이터들의 집합으로, 튜플과 에트리뷰트로 데이터를 정렬하여 관리합니다.

3.2 튜플

- 테이블의 각 행으로 고유한 정보를 담고 있습니다.
- 튜플의 수를 cardinality라고 합니다.

3.3 레코드

- 정보를 처리하는 기본적인 단위로 하나 이상의 항목의 모임을 뜻합니다.
- 개체라고도 하며, 하나의 개체는 여러 개의 속성을 갖습니다.
- 테이블에서 행을 의미합니다.

3.4 속성

- 테이블에서 열을 의미하며 더 이상 쪼갤 수 없는 값을 담고 있습니다.
- 속성의 갯수를 degree라고 합니다.

3.5 필드

- 어떠한 의미를 지니는 정보의 한 조각으로, 데이터베이스 시스템에서 처리의 최소 단위가 됩니다.
- 테이블에서 열을 의미합니다.

3.6 뷰

- 뷰는 다른 테이블을 기반으로 만들어진 가상의 테이블을 의미합니다. 이는 실제로 데이터가 저장되어 있는 것이 아니라, 논리적으로만 존재합니다.
- 논리적 데이터 독립성(각각의 응용 프로그램들이 서로 영향을 받지 않으면서 응용 프로그램이 원하는 논리적 구조를 제공할 수 있는 능력)과 물리적 데이터 독립성(응용 프로그램에 영향을 미치지 않고 데이터의 물리적 구조를 변경할 수 있는 능력)을 제공합니다.

3.7 키

- 데이터베이스에서 조건에 만족하는 튜플을 찾거나 순서대로 정렬할 때 다른 튜플들과 구별할 수 있는 유일한 기준이 되는 Attribute(속성)입니다.
- 기본키, 후보키, 대체키, 슈퍼키, 외래키 등이 있습니다.

3.8 인덱스

- 인덱스는 테이블에 대한 동작의 속도를 높여주는 자료 구조를 말합니다.
- 테이블과 논리적, 물리적으로 독립적이게 존재하면서도 테이블의 데이터를 빠르게 조회할 수 있도록 돕습니다.

4. 1대1(One to One) 관계, 1대다(One to Many) 관계, 다대다(Many to Many) 관계

- 1:1관계는 관련된 필드 모두 기본 키이거나 고유 인덱스가 있는 경우에 만들어집니다.
- 1대다 관계는 관련된 필드 중 한 개만 기본 키이거나 고유 인덱스가 있는 경우에 만들어집니다.
- 다대다 관계는 두 테이블의 외래 키가 기본 키로 구성된 제3의 테이블(병합테이블)에 대한 두 개의 일대다 관계로 만들어집니다.

5. 데이터 무결성(Data Integrity)

- 데이터의 무결성은 데이터의 정확성, 일관성, 유효성이 유지되는 것을 말합니다.
- 데이터 무결성은 개체 무결성, 참조 무결성, 도메인 무결성, 무결성 규칙 등이 있습니다.