1. 요구사항 확인

1) 소프트웨어 개발 방법론

* 소프트웨어 생명주기(SDLC, Software development Life Cycle): 시스템의 요구분석부터 유지보수까지 전 공정을 체계화한 절차

* 생명주기 프로세서

- 요구사항 분석 > 설계 > 구현 > 테스트 > 유지보수

* 생명주기 모델

- 폭포수 모델(Waterful): 가장 오래된, 고전적, 단계별 산출물 명확
- 프로토타이핑 모델(Prototyping): 고객이 요구한 주요기능을 프로토타입으로 구현, 구현 골격
- 나선형 모델(Spiral): 위험 최소화, 점진적,(절차: 계획 및 정의 > 위험분석 > 개발 > 고객평가)
- 반복적모델(Iteration): 병렬적으로 개발 후 통합,

* 소프트웨어 개발방법론(Software Development Methodology): 소프트웨어 개발 전 과정에서 지속적으로 적용할 수 있는 방법, 절차, 기법

* 개발 방법론 종류

- 구조적 방법론(Structed): 하향식, 나씨-슈나이더만 차트 사용
- 정보공학 방법론(Information Engineering): 정보시스템 개발에 필요한 관리 절차, 대형 프로젝트
- 객체지향 방법론(Object-Oriented Development): 객체단위, 객체, 클래스, 메세지 사용
- 컴포넌트기반 방법론(CBD, Compenent Based): 컴포넌트 조립, 기간 단축, 생산성, 확장성, 재사용
- 애자일(Agile): 절차보다는 사람 중심

* 에자일: 절차보다는 사람 중심, 신속 적응정 경량 개발방법론

* 에자일 유형

- XP(eXtreme Programming): 의사소통 개선, 즉각적 피드백

  가치(5가지): 용기, 단순성, 의사소통, 피드백, 존중

  기본원리(12가지): 짝 프로그래밍, 공동 코드 소유, 지속적 통합, 계획, 작은 릴리즈, 메타포어, 간단한 디자인,

  * 테스트기반개발(TDD, Test Driven Development): 테스트 먼주 수행 테스트 통과를 위한 코드

  * 리팩토링(Refactoring): 기능은 바꾸지 않음, 중복제거, 단순화

  40시간 작업, 고객상주, 코드 표준

- 스크럼: 백로그, 스프린트, 스크럼 미팅, 스크럼 마스터, 스프린트 회고, 번 다운 차트

백로그 나눈 후 스크럼 팀 구성, 스프린트 회의, 2~4주단위 스프린트 수행, 수행 중 매일 스크럼 미팅, 스프린트가 끝난후에는 스프린트 회고 수행

- 린: 도요타의 린 시스템 품질기법 적용

* 객체지향 분석 방법론

- 구성요소(클래스 객체 잘 구분)

    클래스: 변수와 메서드를 정의하는 틀

    객체: 클래스에 정의한 것을 토대로 메모리에 할당

    메서드: 객체를 사용하는 방법

    메시지: 객체가 상호 작용을 하기 위한 수단

    인스턴스: 클래스를 통해 만든 실제의 실형 객체

    속성: 객체들이 가지고 있는 데이터 값들을 단위별로 정의

- 객체 지향 기법

    캡슐화: 인터페이스만을 밖으로 드러냄, 정보은닉과 관계

    상속성: 하위클래서에서 재정의 없이 물려받음

    다형성: 오버로딩(매개변수의 유형과 개수가 다르게하여 같은 이름의 메서드)

        오버라이딩(일반 메서드의 구현을 하위 클래스에서 무시하고 재정의)

    추상화: 공통의 성질을 추출하여 추상 클래스 설정

    정보 은닉: 코드 내부데이터와 메서드 숨김, 공개 인터페이스사용

    관계성: 연관화(is-memeber-of), 집단화(is part of, part whole), 분류화(is-instance, 일반화(is-a), 특수화(is-a)

- 설계원칙(SOLID)

    단일책임(SRP, Sigle Reponsibility): 하나의 클래스는 하나의 목적을 위해

    개방폐쇠(OCP, Open Close): 확장 Open, 변경 Close

    리스코프 치환(Liskof Subsititution): 서브타입은 어디서나 자신의 기반타입으로 교체

    인터페이스 분리(Interface Segregation): 사용하지 않는 인터페이스 X

    의존성 역전(Dependency Inversion): 추상을 매개로 메세지 주고받음, 최대한 느슨

* 객체 지향 분석(OOA, Object Analysis): 요구된 문제와 관련된 모든 것 정의하여 모델링

- 종류

    OOSE: 유스케이스에 의한 접근방법. 기능적 요구사항 중심

    OMT: 럼바우: 객체,동적,기증 모델링, 그래픽 표기

    OOD: 설계 문서화 강조

* 기능 모델링

- DFD(Data Flow Diagram): 각 프로세스를 따라 흐르면서 변환되는 모습을 나타넨 그림

- DD(Data Dictionary): 자료 요쇼, 자료 요소들의 집합, 자료의 흐름, 자료 저장소의 의미와 그들 간의

관계, 관계 값, 범위, 단위들을 구체적으로 명시하는 사전

* 프로젝트 관리

- 개념: 기간 내 최소한 비용 사용자 만족 시킴
- 관리 대상: 계획, 품질, 범위
- 3요소: 사람, 문제, 프로세스(People, Problem, Process)

* 비용산정보형

- 하향식: 델파이, 전문가 판단
- 상향식: 세부적인 요구사항과 기능에 따라

    LoC: 각 기능의 원시 코드 라인수의 낙관치, 중간치, 비관치 측정

    Man Month: 한 사람이 1개월 동안 할 수 있는 일

    COCOMO: 프로그램 규모에 따라 조직, 반분리, 임베디드 형으로 나눔

    푸트남: 단계별 인력의 분ㄴ포 가정

    기능점수(FP): 요구 기능을 증가시키는 인자별루 가증치 부여

* 위험 관리 대응 전략:

- 회피Avoidence
- 전가Transerfence
- 완화Mitgation
- 수용Acception

2) 현행 시스템 파악: 형생시스템이 어떤 하위 시스템으로 구성되어 있고 제공 기능 및 연계 정보는 무엇이며 어던 기술요소를 사용하는지를 파악하는 활동

- 절차: 구성기능인터페이스/ 아키텍처 및 소프트웨어 구성/ 하드웨어 및 네트워크
- 소프트웨어 아키텍처: 여러 가지 소프트웨어 구성요소와 그 구성요소가 가진 특성 중에서 외부에 드러나는 특성, 그리고 구성요소 간의 고나계를 표현하는 시스템의 구조나 구조체
- 소프트웨어 아키텍처 프레임워크: 소프트웨어 집약적인 시스템에서 아키텍처가 표현해야하는 내용 및 이들 간의 관계를 제공하는 아키텍처 기술 표준

* 4+1 뷰

- 유스케이스 뷰: 유스케이스 또는 아키텍처를 도출하고 설꼐
- 논리 뷰: 기능적 요구사항 제공 방법
- 프로세스 뷰: 비기능적인 속성 표현
- 구현 뷰: 모듈의 구성

- 배포 뷰: 컴포넌트가 물리적인 아키텍처에 어떻게 배치되는가

* 디자인 패턴 종류

- 생성패턴: Builder, Prototype, Factory Method, Abstract Factory, Singleton
- 구조패턴: Bridge, Decorator, Facade, Flyweight, Proxy, Composite, Adapter
- 행위패턴: Mediator, Interpreter, Iterator, TemplateMethod, Observer, State, Visitor, Command, Stratge, Memento, Chain of Responsibility

* OSI 7계층

- 응용: 데이터 생성
- 표현: 암/복호화
- 세션: 연결 접속및 동기제어
- 전송: 신뢰성 있는 통신 보장
- 네트워크: 최적화된 경로 제공
- 데이터 링크: 전송오류제어
- 물리 전기적 신호 변환

* DBMS 시스템 분석 시 고려사항: 가용성, 성능, 상호 호환성, 기술지원, 구축비용

* 미들웨어 시스템 분석 시 고려사항: 가용성 성능, 기술지원, 구축비용

3) 요구사항 확인

* 요구공학의 개념: 사용자의 요구가 반영된 시스템을 개발하기 위하여 사용자 요구사항에 대한 도출, 분석, 명세, 확인 및 검증하는 구조화된 활동

- 기능적 요구: 입력에 대한 반응, 상황에 대한 동작
- 비기능적요구: 시스템이 갖춰야할 기술, 준수해야할 제한 조건에 관한 기술

* 프로세스

- 도출
- 분석
- 명세
- 확인및 검증(이해했는이 확인(Validation), 요구사항 만족하는지 검증(Verification)

* 확인 및 검증단계의 정형 기술 검토 기법

- 동료 검토: 2~3명이 진행하는 리뷰
- 워크 스루: 오류를 조기에 검출하는 목적
- 인스펙션: 저작자 외의 다른 전문가 또는 팀이 검사하는 공식적인 검사

2. UI

1) UI 요구사항 확인

* UI: 사용자와 시스템 사이에서 의사소통할 수 있도록 고안된 물리적, 가상의 매개체
* UX: 사용자가 직/간접적으로 경험하면서 느끼고 생각하는 총체적 경험
* UI 유형
- C(command)LI: 명령어
- GUI: 펜, 마우스
- N(natural)UL: 신체 부위
- O(organic)UI: 모든 사물
* UI 설계 원칙: 직관성, 유효성, 학습성, 유연성
* UI 품질 요구사항: 기능성, 신뢰성, 사용성, 효율성, 유지보수성, 이식성
* 스토리보드: UI설계를 위한 대부분 정보가 수록된 문서
* 프로토타입: 전체적인 기능을 간략한 형태로 구현한 시제품

2) UI 설계

* UML(Unified Modeling Language): 객체지향 소프트웨어 개발 과정에서 산출물을 명세화, 시각화, 문서화할 때 사용되는 모델링 기술과 방법론을 통합해서 만든 표준화된 범용 모델링 언어
* UML 특징: 가시화 언어, 구축 언어, 명세화 언어, 문서화 언어
* UML 구성요소: 사물(things), 관계(relationship), 다이어그램(Diagram)
* UML 다이어그램
- 구조적,정적 다이어크램: 클래스, 객체, 컴포넌트, 배치, 복합체 구조, 패키지
- 행위적 다이어그램: 유스케이스, 시퀀스, 커뮤니케이션, 상태, 활동, 타이밍
* 목업: 실제 제품이 나오기 전 만드는 모형

3. 데이터 입출력 구현

1) 논리 데이터 저장소 확인

* 데이터 모델 개념: 현실 세계의 정보를 인간과 컴퓨터가 이해할 수 있도록 추상화하여 표현한 모델
- 표시요소: 연산, 구조 제약조건
- 절차
    개념적 설계: 요구에 대한 트랜잭션 설계, 개체관계 다이어그램
    논리적 설계: 트랜잭션 인터페이스 설계, 정규화
    물리적 설계: 성능을 생각한 물리적인 스키마, 반 정규화
* 관계 데이터모델: 데이터를 행과 열루 구성된 2차원 테이블 형태로 구현

- 구성요소: 릴레이션, 튜플, 속성, 카디널리티, 차수, 스키마, 인스턴스
- 관계대수: 원하는 정보와 그 정보를 어떻게 유도하는가, 절차적언어
- 관계 해석: 비절차적언어
* 관계 대수의 연산자 그림으로 따로 정리

* 논리적 데이터 모델링 속성: 개체(Entity), 속성(Atributed), 관계(Relationship)
* E-R 모델: 데이터와 그들 간의 관계를 사람이 이해할 수 있는 형태로 표현
(그려보기)

* 정규화: 관계형 데이터 모델에서 데이터의 중복성 제거
* 이상현상: 정규화를 하지 않은 경우에 대한 이상
    삽입 이상: 불필요한 세부정보를 입력해야 저장되는
    삭제 이상: 원치 않는 정보 삭제
    갱신 이상: 중복 데이터 중 특정 데이터만 업데이트 됨
* 정규화 단계: 원자화, 부분함수 종속제거, 이행함수 종속제거, 결정자 함수 종속제거, 다치종속제거, 조인 종속제거
* 반 정규화: 정규화된 것을 성능 향상과 개발 운영의 단순화를 위해 중복 통합 분리를 수행

2) 물리 데이터 모델 설꼐
* 데이터 무결성: 데이터베이스에 저장된 데이터 값과 그것이 표현하는 현실세계의 실제 값이 일치하는 성질

* 무결성 종류
    개체: 기본키 NOT NULL
    참조: 외래키 참조 키 NOT NULL
    속성: 규칙 준수
    사용자 정의: 의미적 요구사항 준수
    키: 같은 키 X
* 키 특성: 유일성, 최소성
* 키 종류
    기본 키: 실제 사용
    대체 키: 후보키중 사용되지 않은
    후보 키: 키 특성을 만족하는
    슈퍼키: 유일성 만족, 최소성 불만족
    외래키: 참조 데이터 무결성을 위한 제약 조건
* 인덱스: 검색연산을 최적화하기 위한 열에 대한 정보를 구성한 데이터 구조
* 파티셔닝 유형: 레인지, 해시, 리스트, 컴포넌트, 라운드로빈
* 파티션 장점: 성능 향상, 가용성 향상, 백업, 경합 감소

3) 데이터베이스 기초활용
* DB: 다수의 인원 시스템 또는 프로그램이 사용할 목적으로 통합하여 관리되는 데이터의 집합
* DBMS: 데이터 관리의 복잡성을 해결하는 도잇에 데이터 추가, 변경, 검색, 삭제 및 백업, 복구, 보안 등의 기능을 지원하는 소프트웨어
* DBMS 특징: 무결성, 일관성, 회복성, 보안성, 효율성
* 빅데이터: 시스템, 서비스, 조직 등에서 주어진 비용, 시간 내에 처리가 가능한 데이터 범위를 넘어서는 페타바이트 크기의 비정형 데이터(양, 다양성, 속도)
* NoSQL: 전통적인 RDBMS와 다른 DBMS 지칭, 고정된 스키마 X, 조인 연산 X, 수평적 확장
* 데이터 마이닝: 대규모로 저장된 데이터 안에서 체계적이고 자동적으로 통계적 규칙이나 패턴을 찾아내는 기술

4. 통합 구현
1) 연계 매커니즘 구성
- 연계 매커니즘: 연계 모듈 간의 데이터 연계시 요구사항을 고려한 연계방법과 주기를 설계하기 위한 메커니즘

- 직접 연계: 중간 매개체 없음, 소요비용 기간 짧음
- 간접 연계: 서로 상이한 네트워크, 프로토콜 연계 및 통합 가능, 보안 업무 처리 로직 아유롭게 반영

2) 내외부 연계 모듈 구현

* EAI(Enterprise Application Integration): 기업에서 운영되는 서로 다른 플랫폼 및 애플리케이션 간의 정보를 전달, 연계, 통합이 가능하도록 해주는 솔루션
* EAI 유형: 포인트 투 포인트, 허브앤 스포크, 메시지 버스, 하이브리드
* ESB(Enterprise Service Bus): 서로 다른 플랫폼 및 애플리케이션들 간을 하나의 시스템으로 관리 운영할수 있또록 서비스 중심의 통합을 지향하는 아키텍처, 미들웨어 중심, 느슨한 결함
* 웹서비스: 네트워크에 분산된 정보를 서비스형태로 개방, 서비스 지향 아키텍처 개념 실현
    HTTP: POST, GET, PUT 사용
    Hypertext: 링크를 통해 서로 연결된 네트워크 처럼 구성
    HTML: 웹 컨텐츠의 의미와 구조, 마크업 언어
* 웹서비스 유형
    SOAP: HTTP, HTTPS, SMTP 사용 ,xml 기반
    WSDL: 웹 서비스명, 제공 위치, 메시지 포맷 등 상세정보가 기술된 XML 형식
    UDDI: WSDL 을 등록, 검색이 가능한 레지스트리이자 표준
* IPC(Inter-Process Communicaitno): 운영체제에서 프로세스 간 서로 데이터를 주고받기 위한 통신 기술
* IPC 주요기법: 메시지 큐, 공유메모리, 소켓, 세마포어

5. 인터페이스 구현

1) 인터페이스 설계 확인

- EAI, ESB 연계방법 사용

2) 인터페이스 기능 구현

* JSON(Javascript Object Notatino): 속성-값 또는 키-값 쌍으로 이루어진 데이터 오브젝트를 전달하기 위해 인간이 읽을 수 잇는 텍스트를 사용하는 표준 포맷
* XML(Extensible Markup Language): HTML 의 단점을 보오나한 인터넷 언어, SGML의 복잡한 단점을 개선한 특수한 목적을 갖는 마크업 언어
* AJAX(Asynchronous Javascrpit And XML): 자바스크립트를 사용하여 웹 서버와 클라이언트 간 비동기적으로 XML 데이터를 교환하고 조작하기 위한 웹 기술
* REST: 웹과 같은 분산 하이퍼미디어 환경에서 자원의 존재/상태 정보를 표준화된 HTTP 메서드로

주고받는 웹 아키텍처

* 시큐어 코딩 가이드: 입력데이터 검증 및 표현, 보안 기능, 시간 및 상태, 에러 처리, 코드 오류, 캡슐화, API 오용

* DB 보안 적용

- 대칭 키 암호화 알고리즘: 암호화, 복호화에 같은 암호키를 쓰는 알고리즘

- 비대칭 키 암호화 알고리즘: 공개키는 누구나 알 지만, 그에 대응하는 비밀키는 키의 소유자만 알 수 있도록, 공개키와 비밀키를 사용하는 알고리즘

* DB 암호화 기법: API, Plug-in, TDE, Hybrid

* 중요 인터페이스 데이터의 암호화

IPSec: 3계층에서

SSL/TLS: 4계층, 7계층사에서

S-HTTP: 네트워크 트래픽을

3) 인터페이스 구현 검증

* 검증도구 종류

xUnit: jUnit, cppUnit, nUnit

STAF: 재사용 및 확정성

FitNesse: 협업 기능

NTAF: STAF + FitNesse

Selenium: 다양한 브라우저, 개발언어 지원, 기능 테스트를 위한

watir: 루비기반, 모든 언어 브라우저 호환

* 오류 처리방법

- 사용자 화면에서 오류를 인지하도록 구현

- 인터페이스 오류 로그 생성

- 인터페이스 관련 테이블에 오류사항 기록

6. 프로그래밍언어

7. SQL 응용

1) 데이터베이스 기본

* 트랜젝션: 인가받지 않은 사용자로부터 데이터를 보장하기 위해 DBMS가 가져야 하는 특성이자, 데이터베이스 시스템에서 하나의 논리적 기능을 정상적으로 수행하기 위한 작업의 기본 단위

- 원자성: 모두 정상적으로 실행 또는 취소
- 일관성: 고정요소는 그대로

- 격리성=고립성: 트랜잭션이 서로 영향 끼쳐서는 안됨
    - Read Uncommitted
    - Read Committed
    - Repeatable Read
    - Serializable
- 영속성: 트랜잭션 결과는 영원히 저장

* TCL 명령어: COMMIT, ROLLBACK, CHECKPOINT
* DDL정의어
- 대상: 도메인, 스키마, 테이블, 뷰, 인덱스
- 명령어: CREATE, ALTER, DROP, TRUNCATE

- DML: SELECT, INSERT, UPDATE, DELETE

- DCL: GRANT, REVOKE

2) 응용 SQL 작성하기

* 다중행 연산자

IN: 리턴되는 값 중에서 조건에 해당하는 값이 있으면 참

ANY: 서브쿼리에 의해 리턴되는 각각의 값과 조건을 비교하여 하나 이상을 만족하면 참

ALL: 값을 서브쿼리에 의해 리턴되는 모든 값과 조건 값을 비교하여 모든 값을 만족하여야 참

EXISTS: 메인 쿠ᄀᆞ리으 ㅣ비교 조건이 서브쿼리의 결과 중 만족하는 값이 하나라도 존재하면 참

* 집계함수: 여러 행 또는 테이블 전체 행으로부터 하나의 결과값을 반환하는 함수

- 구문: GROUP BY, HAVING

- 종류: COUNT, SUM, AVG, MAX, MIN, STDDEV, VARIANCE

* 그룹함수: 테이블의 젙체 행을 하나 이상의 컬럼을 기준으로 컬럼 값에 따라 그룹화하여 그룹별로 결과를 출력

- ROLLUP: 지정된 컬럼은 소게 등 중간 집계 값을 산출하기 위한 그룹 함수

- CUBU: 결합 가능한 모든 값에 대해 다차원 집계를 생성하는 그룹 함수

- GROUPING SETS: 집계 대상 컬럼들에 대한 개별 집계를 구할 수 있으며, 컬럼 간 순서와 무관한 결과를 얻을 수 있음

* 윈도함수는 OLAP 함수라고도 하며 RANK가 대표적

8. 서버 프로그램 구현

1) 개발환경 구축

* 개발도구 분류

- 빌드도구
- 구현도구
- 테스트도구
- 형상관리 도구

* 하드웨어 개발환경: 웹서버, 웹 어플리케이션 서버, 데이터베이스 서버, 파일서버

* 형상관리 절차

형상식별: ID와 관리번호 부여

형상통제: 형상통제위원회CCB 운영

형상 감사: 무결성 평가

형상 기록: 각종 수해결과 기록

* 형상관리도구

CVS:, 서버와 클라이언트, 다수인원 동시 범용적 운영

SVN: 하나의 서버에서 소스를 쉽고 유용하게 관리

RCS: CVN과 달리, 파일 수정은 한 사람만

BitKeeper: SVN과 비슷, 중앙통제, 대규모 빠름

Git: 알지?

Clear Case: 복수 서버, 복수 클라이언트, 서버 추가 가능

2) 공통 모듈 구현

* 재사용: 개발 시간 및 비용 절감을 위한 최적화 작업

* 재공학: 소프트웨어를 버리지 않음

* 재개발: 기존의 내용을 참조하여 새로 만듬

* 모듈: 그 자체로 하나의 완전한 기능을 수행할 수 있는 독립된 실체

* 모듈화: 소프트웨어의 기능을 향상시키거나 복잡한 시스템의 수정, 재사용, 유지 관리 등이 용이하도록 기능 단위의 모듈로 분해하는 설꼐 및 구현 기법

* 응집도: 모듈의 독립성을 나타내는 정도, 모듈 내부 구성요소 간 연관 정도

우연적: 연관 없음

논리적: 유사한 성격

시간적: 특정 시간에 처리

절차적: 다수의 관련 기능을 순차적 수행

통신적: 동일한 입력과 출력 사용하여 다른 기능

순차적: 모듈 내에서 한 활동으로 나온 출력값을 다른활동이 사용함

기능적: 모든 기능이 단일한 목적을 위해 수행

* 결합도: 모듈 내부가 아닌 외부의 모듈과 연관도 또는 모듈 간의 상호 의존성

내부: 다른 모듈 내부에있는 변수나 기능 사용

공통: 전역 변수 참조

외부: 두 개의 모둘이 외부의 인터페이스 공우

제어: 다른 모듈의 내부 논리조직을 제어하기 위한 목적으로 제어 신호 이용

스탬프: 모듈간의 인터페이스로 배열, 객체, 구조 등이 전달 됨, 동일한 자료구조, 하나 변하면 다 변함

자료: 모듈 간 인터페이스로 전달되는 파라미터를 통해서만 모듈 간 상호작용, 영향 X

* 공통모듈 테스트는 화이트박스 기법을 활용

3) 배치 프로그램: 상호 작용 없이 일련의 작업들을 작업 단위로 묶어 정기적을 ㅗ반복 수행하거나 정해진 규칙에 따라 일괄 처리하는 방법

* 배치 스케줄러: 스프링 배치(대용량), 쿼츠 스케줄러(유연성)

Cron 표현식 계산

10. SW 보안 구축

1) 설계

* 보안 3요소: 기밀성, 무결성, 가용성

* 보안 용어: 자산(Asset), 위협(Threat), 취약점(Vulnerability), 위험(Risk)

* DoS: 시스템을 악의적으로 공격해서 해당 시스템의 자원을 부족하게 하여 원래 의된 용도로 사용하지 못하게함

SYN 플러딩: TCP 프로토콜의 구조적 문제, 서버의 동시 가용자 수를 SYN 패킷만 보 냄

UDP 플러딩: 대량의 UDP 패킷을 만들어 응답메세지를 생상시켜 자원 고갈

스머프/스머핑: 출발지 주소를 공격대상의 IP로 네트워크 전체 ICMP Echo 패킷

PoD: 죽음의 핑, ICMP 패킷(ping)을 아주 크게. 재조합 버퍼의 오버플로우

랜드어택: 출발지와 목적지 IP가 같은 패킷 주소, 가용성 침해

티어드롭: IP 패킷 재조합 과정 잘못된 Fragment Offset

봉크: 패킷 분할 1번을 계속 1번으로 보냄

보잉크: 시퀀스번호를 비정상적으로 설정

* DDos: Dos의 또 다른 형태 여러 대의 공격자를 분산 배치하여 동시에 동작하게 함으로써 특정 사이트를 공격

* 구성요소: 핸들러, 에이전트 마스터, 공격자, 데몬 프로그램

*DRDoS: 공격자는 출발지 IP를 공격대상 IP로 위조하여 다수의 반사 서버로 요청 정보를 전송, 공격대상자는 반사 서버로부터 다량의 응답을 받아서 서비스 거부(DoS)가 되는 공격

* 세션하이제킹(Session Hijacking): TCP의 세션 관리 취약점을 이용한 공격기법

- 비동기화 상태로 패킷이 유실되어 재선송 패킷이 증가

* 애플리케이션 공격: HTTP GET FLOODING, Sloworis, RUDY Attack, SLow Read Attack, Hulk Dos
* 네트워크 공격

    스니핑: 몰래 데이터만 드려다봄

    네트워크 스캐너, 스니퍼: 네트워크 구성의 취약점 탐색 도구

    패스워드 크래킹: 사전(유추함), 무차별(아무거나), 하이브리드(사전 + 무차별), 스푸핑(인증된 시스템인 것처럼 속여)

    ARP 스푸핑: 호스트의 MAC 주소를 자신의 MAC 주소로 위조하여 스니핑

    ICMP Redirect: 특정 목적지로가는 패킷을 공격자가 스니핑

    트로이 목마: 악성 루틴이 숨어있는 겉으로 멀쩡한 프로그램, 실행 시 악성 코드 실행
* 버퍼 오버플로우: 메모리에 할당도니 버퍼크기를 초과하는 야으이 데이터를 입력
* 백도어: 어떤 제품이나 컴퓨터 시스템, 암호시스템 혹은 알고리즘에서 정상적인 인증 절차를 우회
* 주요 시스템 보안 공격기법

    포맷 스트링 공격: 검증되지 않은 입력값

    레이스 컨디션 공격:공유자원 동시 접근할 때

    키로거: 키보드 움직임ㅇ르 탐지

    루트킷: 시스템 침입후 침입사실을 숨김
* 보안 관련용어

    스피어피싱: 사회공학의 한 기법, 위장 메일

    스미싱: SMS 피싱

    큐싱: QR코드 피싱

    봇넷:악의적 컴퓨터들이 네트워크로 연결된 상태

    APT 공격: 지능적 맞춤형 공격

    공급망 공격: 소프트웨어 개발사 네트워크 침입

    제로데이 공격: 알려지지 않은 보안 취약점 악용

    웜: 스스로 복제하여 네트워크를 통해 전파

    악성 봇: 해커의 명령에 의해 원격 제어 실행, 취약점이나 백도어 등을 이용

    사이버 킬체인: APT 공격 방어 분석 모델

    랜섬웨어: 감염된 파일을 암호화 복호화,

    이블 트윈 공격: 무선 Wifi 피싱

    사회공학: 사람들의 심리와 행동 양식을 교묘하게 이용

    트러스트존: 프로세서 안에 독립적인 보안 공격, 하드웨어 기반

    타이포스쿼팅: 유사한 유명 도메인 미리 등록, URL 하이재킹

* 인증기술 유형

지식기반 인증

소지기반 인증

생체기반 인증

특징(행위)기반 기증

* 서버 접근 통제유형

임의적 접근 통제(DAC, Discretionary Access Control): 주체나 그룹의 신분에 근거

강제적 접근 통제(MAC, Mandatory Access Control): 객체에 포함된 정보의 허용등급 과 접근 정보에 대하여 주체가 갖는 접근 허가 권한에 근거

역할 기반 접근 통제(RBAC, Rule Based Access Control): 중앙 관리자가 사용자와 시 스템의 상호관계를 통제

* 3A

Authntication 인증

Authorization 권한 부여

Accounting 계정 관리

* SSO: 커버로스에 사용되는 기술 한 번의 인증 과정으로 여러 컴퓨터상의 자원 이용

* 커버로스: 클라이언트/서버 모델에서 동작 대칭 키 암호기법, 티켓 기반의 프로토콜

* 접근 통제 보호 모델

- 벨-라파듈라: 기밀성 보장
- 비바 모델: 무결성 보장

* 암호 알고리즘 방식

- 대칭 키(양방향 암호방식)

블록 암호 방식: 고정 길이의 블록을 암호화 DES, AES, SEED

스트림 암호 방식: 매우 긴 주기의 난수열, RC4

- 비대칭키(양방향 암호방식, 공개키 암호화 방식)

* RSA: 널리 사용됨, 소인수 분해 이용

* 디피 헬만: 암호키 교환, 공통의 비밀키를 공유

- 일방향 암호방식(해시암호방식): MAC(메시지의 무결성 송신자의 인증), MDC(메시지의 무결성)

* 양방향과 일방향 상세 내용 정리

* 안전한 전송을 위한 데이터 암호화

- VPN(IPSec,SSL)
- IPSEC
- SSL/TSL
- S-HTTP

위의 구분 필요

* 자산목록표: 항목이 존재하고 중요도에 따라 등급을 매긴다

2) 소프트웨어 개발 보안 구현

* 시큐어 코딩 가이드

- 입력 데이터 검증 및 표현
    - XSS: 검증되지 않은 외부 입력데이터가 포함된 웹페이지
    - CSRF: 사이트간 요청 위조, 사용자의 의지와는 무관
    - SQL Injection: 악의적인 SQL문 삽입
- 보안 기능
    - 적절한 인증 없음
    - 부적절한 인가
    - 하드 코드된 비밀번호

시간 및 상태

- 에러 처리
    - 오류 메시지 통한 정보 노출
    - 오류 상황 대응 부재
    - 적절하지 않은 예외 처리
- 코드 오류
    - 널 포인터 역참조
    - 정수를 문자로 변환
    - 주적절한 자원 해제
    - 초기화 되지 않은 변수 사용

캡슐화

- 잘못된 세션에 의한 데이터 정보 노출
- 제거되지 않고 남은 디버그 코드
- 민감한 데이터를 가진 내부 클래스 사용
- 시스템 데이터 정보 노출

세션 통제?

- 불충분한 세션 관리

API 오용

- Null 매개변수 미검자
- DNS LOOkup에 의존
- 위험하다고알려진 함수 사용(strcpy 함수, 글자 수 상관 없이 문자열 복사 가능)

* DRS 유형
    - Mirror Site 실시간 동시 서비스 가능
    - Hot Site 동기, 비동기 방식의 미로링
    - Warm Site 중요성 높은 자원만 보유
    - Cold Site: 데이터만 원격지 보관
* ISMS와 PIMS로 개별 운영되던 인증체계를 하나로 통합한 통합 인증제도 ISMS-P 실행 중

10. 애플리케이션 테스트 관리

1) 애플리케이션 테스트 케이스 작성

* 소프트웨어 테스트 원리

결함 존재 증명

완벽한 테스팅 불가능

초기 집중

결함 집중

살충제 패러독스

정황 의존성

오류-부재의 궤변

* 테스트 산출물

테스트 게획서

테스트 베이시스: 논리적인 Case

테스트 케이스: 테스트를 위한 설계 산출물

테스트 슈트: 시나리오가 포함되지 않은 단순한 테스트 케이스

테스트 시나리오: 테스트 되어야할 기능 및 특징, 테스트가 필요한 상황

테스트 스크립트: 테스트케이스의 실행 순서

테스트 결과서: 테스트 결과를 정리한 문서

* 화이트박스 테스트: 각 으용 프로그램의 내부 구조와 동작을 검사하는 소프트웨어 테스트

구문 커버리지(Statement Coverage): 모든 명령문을 적어도 한 번 수행

결정 커버리지(Decision Coverage): 각 분기의 결정 포인트 내의 전체 조건식 적어도 한번은 참, 거짓 수행

조건 커버리지(Condition Coverage): 각 개별 조건식이 적어도 한 번 참 거짓 수행

조건/결정 커버리지(condition/Decision Coverage): 전체, 개별 조건식 참 거짓 수행

변경 조건/결정(Modified Condtion/ Decision Coverage): 다른 개별 조건식에 영향 X 전체 조건식이 덕립적으로 영향을 주도록 한 조건/결정 커버리지 향상 버전

다중 조건 커버리지(Multiple Coverage)" 모든 개별 조건식의 모든 가능한 조합 100% 보장

기본 경로 커버리지(Base Path Coverage): 수행 가능한 모든 경로

제어 흐름 테스트(Data Flow Testing): 제어프름 그래프에 데이터 사용현황 추가

루프 테스트(Loop Testing): 프로그램의 반복(Loop) 구조에 초점을 맞춰 실시하는 테스트

* 블랙박스 테스트: 프로그램 외부 사용자의 요구사항 명레를 보면서 수행하는 테스트(기능테스트)

동등분할 테스트(Equivalence Partitoinning Testing): 최솟값 바로 위 최대치 바로아래

결정 테이블(Decision Table Testing): 요구사항의 논리, 발생조건을 테이블 나열, 조합

상태 전이 테스트(State Trasition Testing): 한 상태에서 다른 상태로 전이

유스케이스(Use Case Testing): 실제 유스케이스 모델링 있을 때, 테스트케이스명세화

분류트리테스트(Classification Tree MetHod Testing): 트리구조로 분석 및 표현

페어와이즈 테스트(Pairwise Testing): 테스트 데이터 값들 최소한 한 번씩 조합
적은 양에 비해 효율성

원인-결과(Cause-Effect Graph Testing): 그래프 활용, 데이터간 관계출력 분석

비교 테스트(Comparison Testing): 여러 버전 프로그램에 같은 입력 값

오류 추정 테스트(Error Guessing Testing): 개발자가 범할 수 있는 실수 추정

* 검증과 확인의 차이

- 검증(Verification): 소프트웨어 개발 과정을 테스트
- 확인(Validation): 소프트웨어 결과를 테스트

* 테스트 목적에 따른 분류

회복테스트(Recovery Testing): 고의로 실수 유도, 정상적 복귀 여부 확인

안전 테스트(Security Testing): 보안적인 결함 미리 점검

성능 테스트(Performance Testing): 시간 측정

회귀 테스트(Regresstion Testing): 수정 후 새 오류 점검

구조 테스트(Structure Testing): 코드의 복잡도 확인

병행 테스트(Parallel Testing): 변경된 시스템, 기존 테스템 동입 입력값 확인

* 성능 테스트 상세유형

부하 테스트: 부하증가시키펴 임계점 찾기

강도 테스트: 임계정 이상의 부하 가하여 비정상적인 상황에서 동작 확인

스파이크 테스트: 짧은 시간 사용자 몰릴 때 반응 측정

내구성 테스트: 오랜 시간 시스템에 높은 부하 테스트

* 명세 기반 = 블랙 박스
* 구조 기반 = 화이트 박스
* 경험 기반 = 블랙박스(탐색적 테스트, 오류 측정)
* 테스트 커버리지: 테스트 수행정도를 나타내는 값

기능 기반 커버리지

라인 커버리지

코드 커버리지

* 테스트 케이스: 특정 요구사항을 준수하는 지를 확인 위해 개발된 입력 값, 실행 조건, 예상된 결과의 집합
* 테스트 오라클: 테스트의 결과가 참인지 거짓인지 판단하기 위해서 사전에 정의된 참값을 입력하여

비교하는 기법

참 오라클: 모든 입력값에 기대하는 결과를 생성

샘플링 오라클: 특정한 몇개의 입력값에 대해서만

휴리스팅 오라클: 샘플링 개선, 특정함 입력값의 올바른 결과를 제공, 나머지 값들은 추정(Heuristic)

일관성(Consistent) 오라클: 애플리케이션 변경이 있을 때 수정 전과 후 확인하는오라 클

* 테스트 레벨: 함께 편성되고 관리되는 테스트활동의 구룹

단위 테스트: 단위 모듈, 서브루틴 등

통합 테스트: 모듈 사이의 인터페이스, 통합된 컴포넌트 간의 상호작용

시스템 테스트: 통합도니 단위 시스템의 기능이 시스템에서 정상적으로 작동하는가

인수 테스트: 계약상의 요구사항이 만족되었는가

* 테스트 시나리오: 테스트 수행을 위한 여러 테스트 케이스의 집합, 테스트케이스의 동작 순서를 기술한 문서

2) 테스트 수행

* 테스트 드라이버: 모듈 테스트 수행 후 결과를 도출하는 시험용 모듈, 하위 모듈 호출하는 상위 모듈 역할

* 테스트 스텁: 일시적 필요한 조건만을 가지고 임시로 제공되는 시험용 모듈, 상위 모듈에서 호출되는 하위 모듈의 역할

* 하향식통합에서 최하위 모듈은 깊이-우선 또는 너비-우선 방식으로 통합

* 테스트 하네스: 애플리케이션 컴포넌트 및 모듈을 테스트하는 환경의 일부분, 테스트를 지원하기 위한 코드와 데이터, 개발자가 작성

- 구성요소 드라이버, 스텁, 수트, 케이스, 시나리오, 스크립트, 목 오브젝트

3) 성능 개선

* 애플리케이션 성능 지표: 처리량, 응답시간, 경과시간, 자원 사용률

* 배드코드

외계인 코드: 아주 오래된, 참고문서 없음

스파게티 코드: 가독성 떨어지는 코드

알 수 없는 변수 명

로직 중복

* 클린 코드 원칙: 가독성, 단순성 의존성 최소, 중복성 제거, 추상화

* 리팩토링: 유지보수 생상성 향상을 목적으로 기능을 변경하지 않고, 복잡한 소스 코드를 수정, 보완하여 가용성 및 가독성을 높이는기법

11. 응용 S,W 기초 기술활용

* 운영체제: 사용자가 컴퓨터의 하드웨어를 쉽게 사용할 수 있또록 인터페이스를 제공해주는 소프트웨어

- 종류

    윈도우즈: GUI, 선점형 멀티태스킹, 자동감지, ILE

    유닉스: 90%이상 C언어, 범용 다중 사용자 방식의 시분할 운영체제

    리눅스: 유닉스 기반, 데비안, 레드헷, Fedro, Ubunto,등 다양하게 출시

    Mac: 애플,

    Android: 리눅스 기반, 자바 코틀린,

* 운영체제 명령어

* 메모리 배치 기법

    최초적합(First Fit): 프로세스가 적재될 수 있는 가용 공간 중 첫 번째 분할에 할당

    최적적합(Best Fit): 가용 공간 중 가장 크기가 비슷한 공간

    최악적합(Worst-Fit): 가용 공가 중 가장 큰 공간

* 교체기법

FIFO

LUR

LFU

OPT

NUR

SCR

* 선점형 스케줄링 알고리즘

RR, 라운드로빈: 프로세스는 같은 크기의 CPU 시간을 할당, 시간내 미완료시 뒤로

SRT, 언제라도 가장 짧은 시간 소요되는 프로세스 먼저 수행

MLQ, 다단계 큐: 작업을 분할, 상위 단계 작업에 잃나 하위 단게 작업이 선점

* 비선점형 스케줄링 알고리즘

Priority

DeadLine

FCFS: 도착 순서에 따라

SJF: 프로세스 도착 시점에 따라 가장 짧은 서비스 시간을 프로세스가 종료시까지 자 원 점유, 기아 현상

HRN, 응답률이 가장 높은 것을 선택, 기아현상 해결

* 가상화: 물리적인 리소스들을 사용자에게 하나로 보이게하거나, 하나의 물리적인 리소스를 여러개로 보이게하는 기술

* 클라우드 컴퓨팅: 인터넷을 통해 가상화된 컴퓨터 시스템 리소스를 제공, 정보를 자신의 컴퓨터가 아닌 클라우드에 연결된 다른 컴퓨터로 처리하는 기술

- 클라우드 컴퓨팅 종류

사설 클라우드: 기업 또는 조직 내부에 구축된 클라우드, 보안성 높음

공용 클라우드: 다중 사용자를 위한 클라우드, 비용 지불

하이브라우드 클라우드: 사설 + 공용

- 클라우드 컴퓨팅 유형

IaaS: 서버, 스토리지 같은 시스템 자원을 제공

PaaS: 애플리케이션을 개발, 실행, 관리할 수 있는 플랫폼 제공

SaaS: 웹 브라우저에 접속하여 소프트웨어를 서비스 형태로 이용하는 서비스

2) 네트워크

* 네트워크: 원하는 정보를 원하는 수신자 또는 기기에 정확하게 전송하기 위ㅜ한 기반인프라

WAN 광대역

LAN 근거리

* 계층별 장비

- 1계층: 허브, 리피터
- 2계층: 브리지, NIC, 스위칭 허브, L2 tmdnlcl
- 3계층: 라우터, 게이트웨이, 망 스위칭 허브
- 4계층: L4 스위치

* 프로토콜: 서로 다른 시스템이나 기기들 간의 데이터 교환을 원할히 하기 위한 표준화된 통신규약

- 프로토콜 3요소

    구문(Syntax): 데이터 형식, 코딩, 신호 레벨 등의 규정

    의미(Semantic): 조정과 에러처리를 위한 규정

    타이밍(Timing): 속도조절과 순서 관리 규정

- 네트워크 프로토콜: 메시지를 주고받는 양식과 규칙의 체계

* 데이터 링크계층: 링크의 설정과 유지 및 종료 담당

* 네트워크 계층: 다양한 길의 패킷 전달, QOS를 위한 수단 제공

* IPv4: 32비트, (헤더 20바이트 이상),

*IPv6: 128비트(헤더 40바이트 이상)

* 라우팅 프로코톨(3계층)

- 내부라우팅프로토콜(IGP): RIP, IGRP, OSPF, EIGRP

    RIP: 벨만-포드 알고리즘

    OSPF: 다익스트라알고리즘

- 외부라우팅프로토콜(EGP): BFP

* TCP

- 특징

    신뢰성보장

    연결 지향적 특징

    흐름제어

    혼잡제어

* UDP: 비연결성, 신뢰성 없으며, 순서화되지 않은 데이터그램 서비스를 제공하는 전송계층 통신프로토콜

* 시간남으면 TCP, UDP 헤더구조 살펴보기

* 세션계층 프로토콜: RPC, NetBIOS

* 표현계층 프로토콜: JPEG, MPEG

* 응용계층 프로토콜: HTTP, FTP, SMTP, POP3, IMAP, Telnet, SSH, SNMP

* 애드 혹(ad-hoc) 네트워크: 노드들에 의해 자율적으로 구성되는 기반구조가 없는 네트워크

    완전 독립형 가능

    긴급 구조, 긴급회의, 전쟁터에서 사용됨

* 네트워크 설치구조

- 버스형: 하나 네트워크, 여러 노드

- 트리형: 노드가 계층적

- 링형: 모든 노드가 하나의 링에 순차적으로 연결

- 성형: 허브에 점대 점 연결

* 네트워크 관련 신기술 용어
- SDN: 네트워크 관리자를 위한 기술
- NFV: 범용 하드웨어에 가상화 기술 적용
- Wi-SUN: 스마트 그리드와 연계
- NFC: RFID 확장 기술
- 스몰 셀: 소형 기지국
- 블루투스: 2.4hz 아시죠?
- BLE: 저전력 기반 기기 간 근거리 무선 통신 기능을 제공하는 기술
- Zing: 키오스크에 사용된는 NFC
- BcN: 멀티 실시간 보장하는 광대역 통합망
- C-V2X: 이동통신 기술 기반, 자율주행자동차를 위한 기술
- 메시 네트워크: 소수국 사이 대용량 처리 유리
- UWB: 저전력, 데이터 전송
- UsN: 통신 기능있는 스마트 RFID 태그 및 센서, 환경정보 탐지 네트워크 전송
- WBAN: 3M이내 무선 네트워크
- NDN: 기존 IP 주소 대신 Data 이름활용
- 네트워크 슬라이싱: 5G 슬라이싱
- NOMA: 비직교 다중 접속 기술
- MEC: 모바일 코어 망의 혼잡 완화하는 기술
- IOT: 사물에 있는 인터넷
- MQTT: IoT에 사용되는 프로토콜, publish, subscribe방식의 경량 메시징 전송
- COAP: 비동기 전송, REST 기반 프로토콜
- Zigbee: 저비용 저속, 저전력 근거리 무선 통신
- 스마트 그리드: 고품질 전력서비스 제공, 에너지 이용효율 극대화
* Haddop: 디비관련 신기술 용어, 오픈소스 기반 분산 컴퓨팅 플랫폼, 자바 소프트웨어 프레임워크
3) 기본 개발환경 구축하기
* 인프라 구성방식
- 온프레미스: 외부 인터넷망 차단, 인트라넷만 활용
- 클라우드 방식: 클라우드 공급 서비스를 하는 회사들의 서비스를 임대
- 하이브리드: 온프레미스 + 클라우드
* RAID: 하나의 대형 저장 장치 대신 다수의 저용량 저자장치를 배열로 구성
- 단계 구분 키워드

0 패리티 없는 스트라이핑

1 패리티 없는 미러링

2 오류정보부호 기록하는 하드웨어

3 바스크에 바이트 단위, 패리티 정보 따라보관하는 하드웨어

4 디스크에 블록단위, 패리티 따로

5 패리티 배분, 적어도 3개 디스크

6 패리티 배분, 적어도 4개 디스크

12. 제품 소프트웨어 패키징

* 릴리즈 노트 작성 항목

- 헤더: 문서 이름
- 개요: 제품 및 변경에 대한
- 목적: 새로운 기능, 버그 수정
- 이슈 요약: 버그의 간단한 설명, 또는 릴리즈 추가항목 요약
- 재현 항목: 바그 발견에 따른 재현 단계 기술
- 수정 개선 내용:
- 사용자 영향도:
- 소프트웨어 지원 영향도: 지원 프레세스 및 영향도
- 노트: 설치 항목, 업그레이드 항목 메모
- 면책 조항: 복제방지, 회사 제품 관련 메세지, 중복 등 참조에 대한 고지사항
- 연락 정보

* 디지털 저적권 관리(DRM): 중앙의 클리어링 하우스에서 컨텐츠 제공자, 분배자, 소비자 간의 패키징 배포 및 키 관리, 라이선스 발급 관리를 수행한다.

DRM 구성요소

콘텐츠 제공자: 저작관자

콘첸트 소비자: 사는 사람

콘텐츠 분비자: 암호화된 콘텐츠를 유통하는 곳이나 사람

클리어링 하우스: 사용 권한, 라이센스 발급, 사용량에 따른 관리를 수행, 키 관리,

DRM 콘첸츠: 암호화된 콘텐츠, 메타데이터, 사용정보 패키징하여 구성된 콘텐츠

패키저: 가능한 단위로 묶는 도구

DRM 컨트롤러: 배포된 디지털 콘텐츠의 이용권한 통제

보안 컨테이너: 전자적 보안 장치

* 메뉴얼 작송 항목

- 목차 및 개요
- 문서 이력 정보
- 설치 매뉴얼 주석
- 설치 도구의 구성

* 버전 관리 도구를 통한 관리작업: 버전 관리 백업 및 복구, 동일 버전 공동작업, 여러 버전 솔루션 작업
* 백업 유형: 전체 백업, 차등백업(마지막 백업 이후 변경된거 백업), 증분백업(정해진 시간 백업)
* 백업 정책: 디스크 1일 1회, cd 1주일 1회, 최소 D-2일분 보관