| 대분류/20<br>정보통신 | 중분류/01<br>정보기술 | 소분류/02<br>정보기술개발 | 세분류/02<br>응용SW엔지니어링 | 능력단위/12 |
|---|---|---|---|---|

NCS학습모듈

# 인터페이스 구현

LM2001020212_16v4

교육부

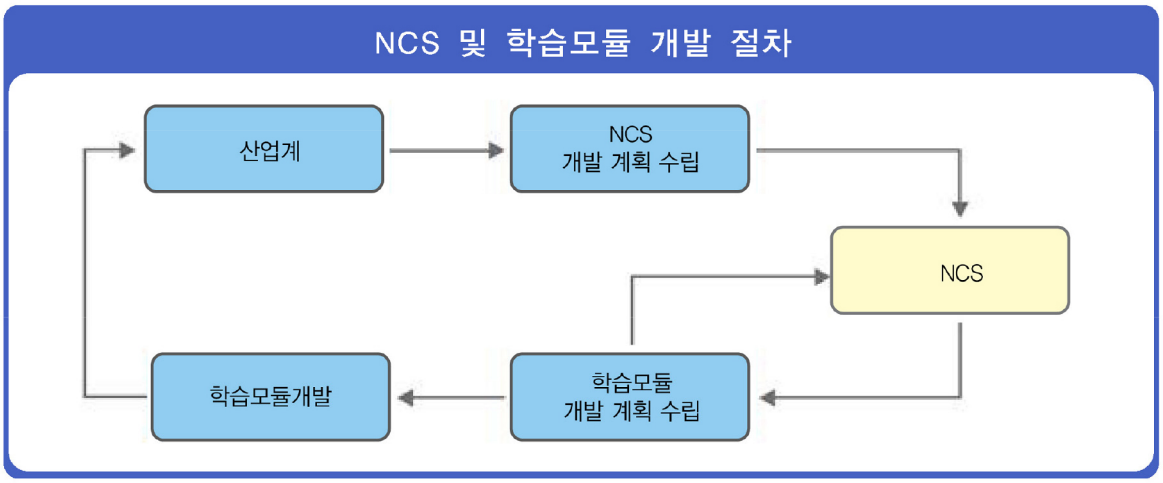
# NCS 학습모듈의 이해

※ 본 학습모듈은 「NCS 국가직무능력표준」 사이트(http://www.ncs.go.kr) 에서 확인 및 다운로드 할 수 있습니다.

## (1) NCS 학습모듈이란?

- 국가직무능력표준(NCS: National Competency Standards)이란 산업현장에서 직무를 수행하기 위해 요구되는 지식·기술·소양 등의 내용을 국가가 산업부문별·수준별로 체계화한 것으로 산업현장의 직무를 성공적으로 수행하기 위해 필요한 능력(지식, 기술, 태도)을 국가적 차원에서 표준화한 것을 의미합니다.

- 국가직무능력표준(이하 NCS)이 현장의 '직무 요구서'라고 한다면, **NCS 학습모듈은 NCS의 능력단위를 교육훈련에서 학습할 수 있도록 구성한 '교수·학습 자료'입니다.** NCS 학습모듈은 구체적 직무를 학습할 수 있도록 이론 및 실습과 관련된 내용을 상세하게 제시하고 있습니다.

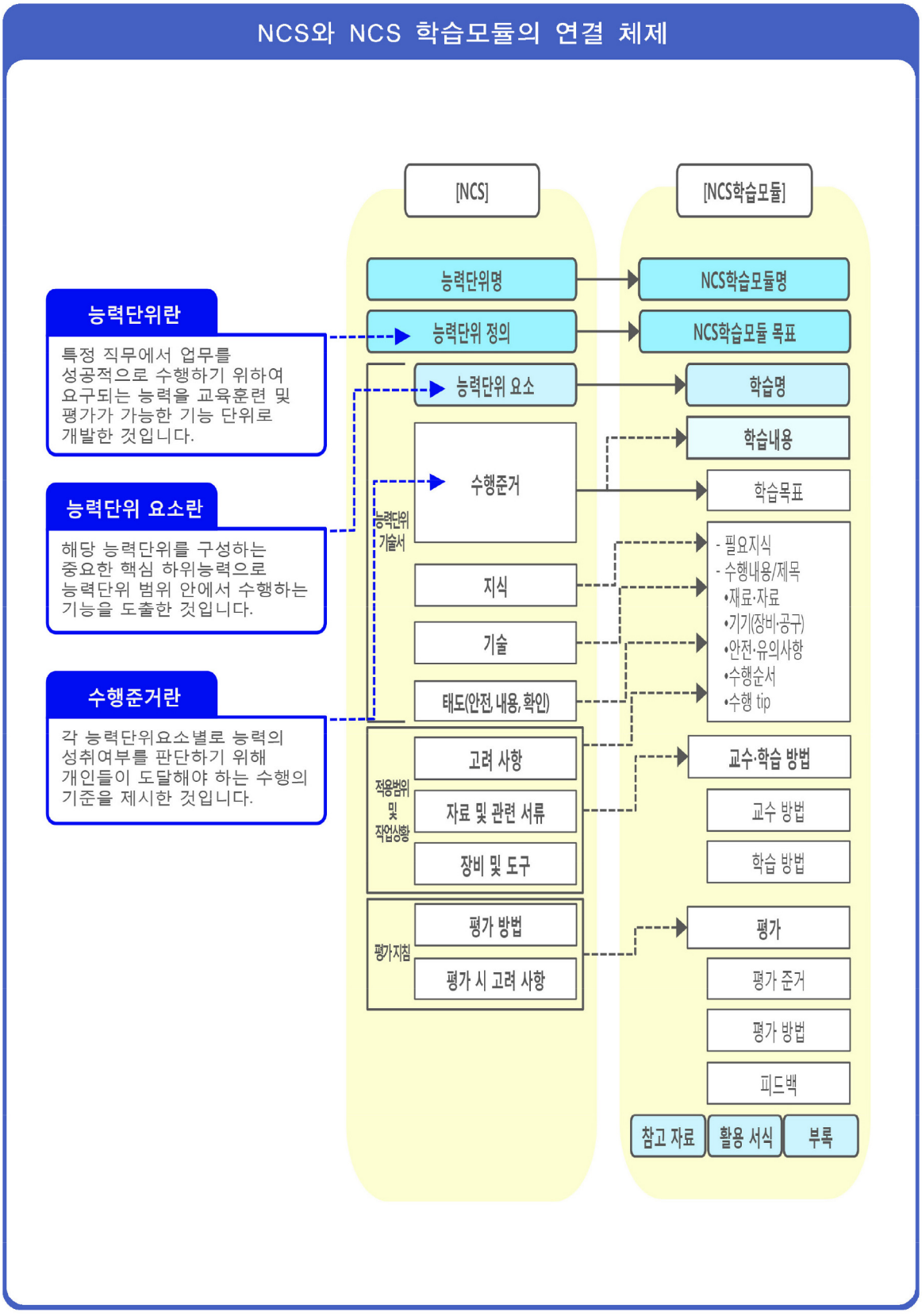



- **NCS 학습모듈은 다음과 같은 특징을 가지고 있습니다.**

  첫째, NCS 학습모듈은 산업계에서 요구하는 직무능력을 교육훈련 현장에 활용할 수 있도록 성취목표와 학습의 방향을 명확히 제시하는 가이드라인의 역할을 합니다.

  둘째, NCS 학습모듈은 특성화고, 마이스터고, 전문대학, 4년제 대학교의 교육기관 및 훈련기관, 직장교육기관 등에서 표준교재로 활용할 수 있으며 교육과정 개편 시에도 유용하게 참고할 수 있습니다.

● NCS와 NCS 학습모듈 간의 연결 체제를 살펴보면 아래 그림과 같습니다.



| | |
|---|---|
| 기능 모델링 | 기능 모델링 |
| 애플리케이션 설계 | 애플리케이션 설계 |
| 정적 모델 설계 | 정적 모델 설계 |
| 동적 모델 설계 | 동적 모델 설계 |
| 화면 설계 | 화면 설계 |
| 화면 구현 | 화면 구현 |
| 애플리케이션 테스트 관리 | 애플리케이션 테스트 관리 |
| 애플리케이션 테스트 수행 | 애플리케이션 테스트 수행 |
| 소프트웨어 공학 활용 | 소프트웨어 공학 활용 |
| 소프트웨어 개발 방법론 활용 | 소프트웨어 개발 방법론 활용 |

## 수행 내용 / 내부, 외부 모듈 간 공통 기능 및 데이터 인터페이스 확인하기

**재료 · 자료**

- 인터페이스 설계서(정의서), 정적·동적 모형 설계도
- 애플리케이션, 네트워크, 보안, 서버 구성 지식 및 자료
- 시스템에 대한 요구 사항 정의서, 아키텍처 구성도
- 정보 전략 계획(ISP: Information Strategy Planning), 정보 시스템 마스터플랜 (ISMP: Information System Master Plan), 제안 요청서(RFP: Request For Proposal) 시스템 개발을 위한 유관 참고 산출물

**기기(장비 · 공구)**

- 전산 장비: 인터넷, 컴퓨터, 프린터, 복사기, 빔 프로젝터 등
- 지원 도구: 문서 작성 도구, 문서 발표 도구

**안전 · 유의 사항**

- 학습자는 학습 전에 강의실 또는 실습실의 안전·유의 사항을 충분히 숙지한다.
- 학습장 안전 수칙 및 운영에 관한 사항을 학습장의 특성에 맞게 비치하고 게시하여야 한다.
- 실험·실습은 정해진 방법과 절차에 따라 실시한다.
- 학습을 통하여 도출된 결과는 반드시 재평가하여 피드백한다.
- 각 시스템에 대한 공통 기능 도출 및 인터페이스 개발은 주관적일 수 있으니 토의를 통하여 자신의 산출한 결과에 대하여 상호 검증을 받는다.
- 토의를 진행할 때에는 비판적인 자세보다는 더 나은 생각을 개진하는 생산적인 방향으로 진행하여야 하며, 상대방의 비판 및 의견을 열린 마음으로 수용하는 자세가 중요하다.

**수행 순서**

1 인터페이스 설계서의 외부 및 내부 모듈의 기능을 확인한다.

인터페이스와 통신하는 외부 및 내부 모듈의 기능을 인터페이스 설계서를 통하여 확인하여 향후 인터페이스 개발에 필요한 공통 영역을 도출할 수 있다.

| 구분 | 파라미터 (변수) | 데이터 표준 | 확인 출처 |
|---|---|---|---|
| | 전표 계정 정보 | - 급여 계정, 상여 계정, 소득세, 주민세, 국민연금, 건강 보험, 고용 보험, 회사 공제 금액 계정, 지급 은행 코드, 지급 계좌<br>- 각 계정은 회사 회계 계정에 맞게 코드화 입력[급여 계정 ABC3343F(예시 코드)]<br>- 은행 코드는 임의 자리로 정의(XX 은행: KBY)<br>- 지급 계좌는 - 없이 숫자로 입력 | 데이터 인터페이스 |
| 출력 정보 | 전표 일반 정보 | - 발생 시기: YYYYMMDD 형식<br>- 전표구분: 매입 AP 매출 AR<br>- 세금 신고 여부: TEX_YN<br>- 보조 정보를 통한 전표의 설명: CHAR 형태 500자 이내 | 데이터 인터페이스 |
| | 전표 금액 | - 각 부서별 지급, 공제 항목에 대한 금액: unsigned Int (-, 소수점 자리 없음)<br>- 개인별 지급 금액: unsigned int | 데이터 인터페이스 |
| | 차변, 대변 | - 각 계정별 마스터 정보 발생 (급여, 상여 세금, 사내 공제 항목)<br>- 디테일 정보(각 귀속 부서별 급여 sum) 발생<br>* 데이터 포맷은 차후 엑셀 정의 | 데이터 인터페이스 |
| | 차·대변 검증 | - 계정별, 부서별 항목 수와 총금액 검증<br>예시) 부서별 검증<br>급여 계산값: 부서 XXX 인원 ○○명 금액 3333<br>전표 발생: 부서 XXX 인원 ○○명 금액 3333<br>XXX 부서 검증 결과: 정상 발생 | 인터페이스 기능 |
| | 거래처 정보 | - 거래처(직원): 사번_이름 형태로 정의<br>- 계좌 정보: 은행 코드_계좌 번호로 정의 | 데이터 인터페이스 |

**수행 tip**

- 데이터 인터페이스와 인터페이스 기능이 일치하는지 검토하고 필요시 인터페이스 데이터를 추가 또는 조정한다.
- 예상 시나리오를 다양하게 생각하고 토의하여 예외 사항까지 모두 담을 수 있도록(빠지는 것이 없도록) 토의한다.

## 학습 1 교수 · 학습 방법

### 교수 방법

- 인터페이스 설계서(정의서, 정적·동적 모형 등)의 목적, 유형, 각 요소 및 특징에 대해서 설명하고 이를 분석할 수 있도록 토의를 진행한다.
- 인터페이스 설계서의 내부 및 외부 모듈 개념에 대해서 설명하고 이를 토의를 통해 식별하고 검증할 수 있도록 한다.
- 각 기능을 통해 데이터 인터페이스 및 이를 구성하는 입력값, 출력값을 확인하도록 토의를 진행한다.
- 시나리오 형태로 외부 및 내부 모듈을 포함한 전체 서비스의 필요 기능을 작성하고 이와 연계된 기능 및 인터페이스 기능을 도출하기 위해 토의를 진행한다.
- 데이터 인터페이스와 인터페이스 기능을 참고로 데이터 인터페이스 표준을 작성하고 이에 대해서 상호 검증을 진행한다.

### 학습 방법

- 인터페이스 설계서(정의서, 정적·동적 모형 등)의 목적, 유형, 각 요소 및 특징에 대해서 이해하고 이를 분석한다.
- 인터페이스 설계서의 내부 및 외부 모듈 개념을 이해하고 각 모듈의 기능을 이해하여 이해한 내용을 상호 검증한다.
- 전체 시나리오를 고려하여 각 기능을 통해 데이터 인터페이스 및 이를 구성하는 입력값, 출력값을 분석한다.
- 시나리오 형태로 외부 및 내부 모듈을 포함한 전체 서비스의 필요 기능을 분석하고 이와 연계된 기능 및 인터페이스 기능을 도출한다.
- 데이터 인터페이스와 인터페이스 기능을 참고로 데이터 인터페이스 표준을 분석하고 작성한 후에 상호 검증을 통하여 빠짐없이 데이터 인터페이스를 도출한다.

구현을 정의하고 이를 분석하여 기능 구현에 활용한다.

1. 기능 구현의 정의 내용을 분석한다.

상세하게 정의된 기능 구현 정의 내용을 토대로 어떻게 구현할 것인지 분석한다. 분석 시 구현 방법, 범위, 구체성 정도까지 고려하여 분석 후 빠르고 정확한 구현이 될 수 있도록 준비한다.

<표 2-7> 정의된 인터페이스 기능 구현 분석

| 기능 구현 정의 | 상세 기능 | 분석 결과 |
|---|---|---|
| 1. 송신 측에서 필요 대상을 선택하여 관계사에 데이터를 전송한다. | (1) 대상자 선택 | DB에 저장된 인사 정보를 SQL로 선택하여 인터페이스 대상을 확인한다. |
| | (2) 인터페이스 데이터 생성 | JSON 또는 인터페이스 테이블 형식에 맞게 DB에 있는 정보를 가공한다. |
| | (3) 인터페이스 데이터 전송 요청 | 인터페이스 데이터를 생성 후 해당 데이터를 http에서 인터페이스 또는 수신 측으로 전송한다. |
| 2. 필요 항목에 대해서 데이터를 전달한다. | (1) 인터페이스 데이터 전송 | 수신지 주소에 REST 방식으로 전달될 수도 있으며 DB Connection을 이용하여 DB Prodecure, Trigger 등을 활용, 전송한다. |
| | (2) 인터페이스 데이터 수신 | 인터페이스 데이터를 수신하고 데이터의 형식이 맞는지 점검한다. 각 Rule을 사전에 정의하고 이를 활용한다. |
| | (3) 인터페이스 데이터 파싱 | 인터페이스 객체를 파서를 이용하여 파싱하거나 인터페이스 테이블에서 SELECT 한다. |
| | (4) 인터페이스 데이터 검증 | 각 세부 데이터의 포맷과 논리적인 제약조건에 걸리지 않는지 검사한다(사용자가 정의한 룰, 예: salary는 0원도다 많아야 한다.). |
| | (5) 후속 기능 수행 | 각 세부 데이터를 활용해서 수신 시스템에서 정의된 Action을 진행한다. |
| 3. 전달 후 전달받은 관계사로부터 전달 여부값을 반환받는다. | (1) 인터페이스 수신 결과를 생성, 전송 | 정상적으로 신되면 true, 아니면 false의 return값을 JSON 형태로 전송한다. |
| | (2) 수신 측에서 처리한 인터페이스 결과를 반환받음. | 수신 측에서 보낸 응답 JSON을 통해 수행 여부를 확인하고 이를 화면상으로 표시한다. |

2 인터페이스를 구현한다.

분석된 기능 구현 정의를 바탕으로 인터페이스 기능을 구현한다(데이터 통신을 사용하는 인터페이스와 인터페이스 엔티티를 활용한 기능 구현을 모두 다루어 본다.).

1. 데이터 통신을 사용하는 인터페이스를 구현한다.

최근에 경량화 되고 가장 많이 사용하는 인터페이스 방법인 JSON을 이용하여 인터페이스를 구현한다.

(1) 인터페이스 객체 생성까지를 구현한다.

인터페이스 객체를 생성하고 전송하기 위해 인터페이스 객체를 생성할 데이터를 선택한다. 선택한 데이터는 JSON을 이용하여 인터페이스 객체를 만드는 데 사용한다.

(가) 인터페이스 객체를 생성하기 위한 데이터를 선택한다.

인터페이스를 객체를 생성할 데이터를 각 시스템 및 환경에 맞게 선택한다. 일반적으로 데이터베이스에 있는 정보를 sql을 통하여 선택한 후 이를 JSON으로 생성한다.

<표 2-8> DB에서 sql로 데이터를 가져오는 예시(JAVA 기준)

**예시**

```
Statement stmt; // sql을 실행하는 자료 구조
ResultSet rs; // 실행한 sql을 저장하는 자료 구조
String sql; sql을 작성할 변수
sql = "    SELECT EMPNO, NAME, JIKGUB, ORG_NM ..
        \n FROM EMP_HISTORY
        \n WHERE EMPNO = 'XXX' --사번
        \n     AND HISTORY_NO = 'DDD' -- 발령 번호"
try{
  rs = stmt.executeQuery(sql);  // sql 실행 rs에 저장
}catch(Exception e){
 throw e;
}
return rs; //실행 결과 return
```

(나) 인터페이스 객체를 구현한다.

sql 실행 결과를 저장한 객체를 화면으로 가져와서 JSON 객체를 생성한다. JSON 형식에 맞게 javascript 객체에 저장하면 된다.

<표 2-9> JSON 인터페이스 객체를 만드는 예시(jquery 사용)

**예시**

```
JSONObject obj = new JSONObject();
obj.put("EMONP",${rs.EMPNO});  //jquery로 데이터를 매핑
obj.put("NAME",${rs.NAME});
....
```

### 수행 tip

- 인터페이스 각 단계와 구현 방법을 이해하고 단계별로 예외사항 발생 시 대응 방안을 정의한다.
- 사용자에게 예외 사항 유형과 원인을 알려 주어야 하며, 동일 예외 사항이 발생되지 않도록 프로그램을 개선해야 한다.

# 2-4. 인터페이스 보안 기능 적용

| 학습 목표 | • 응용소프트웨어와 관련된 내외부 모듈 간의 연계 데이터의 중요성을 고려하여 인터페이스 보안 기능을 적용할 수 있다. |
|---|---|

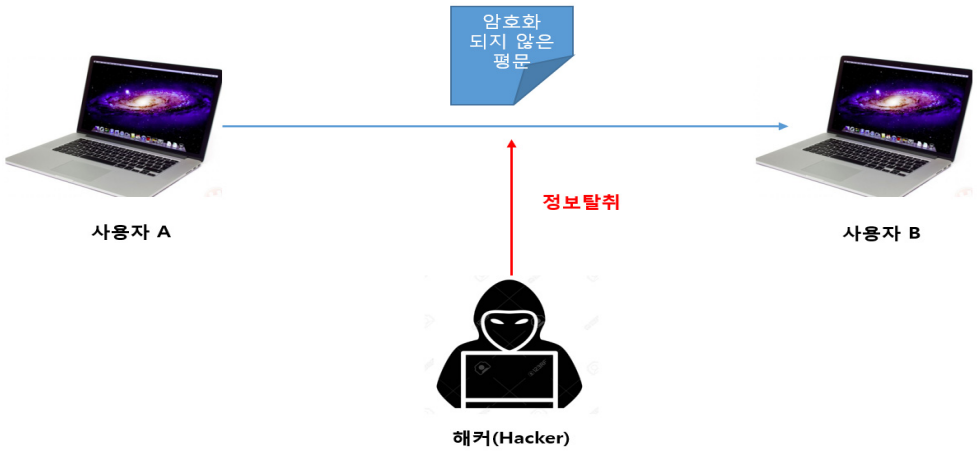
## 필요 지식 /

### 1 구현된 인터페이스의 주요 보안 취약점

인터페이스는 시스템 모듈 간 통신 및 정보 교환을 지원하므로 데이터 변조·탈취 및 인터페이스 모듈 자체의 보안 취약점이 있을 수 있다.

#### 1. 데이터 통신 시 데이터 탈취 위협

데이터 통신 내역을 중간에서 감청하여 기밀성을 훼손할 수 있는 기법으로 스니핑(Sniffing)이라고 불리기도 한다. 스푸핑(Spoofing)처럼 공격 대상을 직접적으로 속이는 해킹이 아닌 수동적(Passive) 해킹 공격 기법이며, 도청할 수 있도록 중간에 설치되는 도구를 스니퍼(Sniffer)라고 한다. 주로 패킷 분석기 같은 툴을 통해서 진행된다.




[그림 2-6] 스니핑 공격 개념도

#### 2. 시큐어 코딩

대표적인 웹 애플리케이션의 보안 취약점 발표 사례인 OWASP(Open Web Application Security Project) Top 10을 참고하여 KISA(한국 인터넷 진흥원)에서 SW 보안 약점 가이드를 발표하였고 SW 보안 취약점, 약점 및 대응 방안이 구체적으로 서술되어 있다.

**수행 순서**

1 인터페이스의 보안 취약점을 분석한다.

인터페이스 구현이 어떻게 되어 있는지를 분석하고 각 구간에 어떤 보안 취약점이 있는지를 다양한 관점에서 분석한다.

1. 인터페이스 각 구간의 구현 현황을 분석한다.

인터페이스 기능이 수행되는 각 구간이 어떻게 구현되어 있는지를 분석한다.

(1) 인터페이스 송수신부의 구현 현황을 분석한다.

인터페이스 보안 취약점을 분석하기 위해 송수신 영역의 구현 기술 및 특징을 구체적으로 분석한다. 송수신 영역의 구현 기술은 보안 취약점이 발생할 수 있는 영역을 구분하여 분석한다.

<표 2-21> 인터페이스 송수신부의 구현 현황 분석 예시(인터페이스 객체 전송 예시)

| 구분 | Application 영역 | DataBase 영역 | Network 영역 |
|---|---|---|---|
| 송신 데이터 선택 | APP 데이터를 Application 영역에서 선택한다. | 데이터베이스에서 송신 데이터를 선택할 수 있다. | Application과 Database 간 DB Connection을 진행한다. |
| 송신 객체 생성 | JSON, XML 등 객체를 생성한다. | | |
| 인터페이스 송신 | 생성된 객체를 수신 측으로 송신한다. | | 송신 측과 수신 측의 네트워크를 연결한다. |
| 인터페이스 수신 | 생성된 객체를 수신하고 파싱한다. | | |
| 데이터 처리 결과 전송 | 파싱한 데이터를 처리하고 최종 결과를 반환한다. | 선택된 데이터를 처리하고 최종 결과를 반환한다. | Application과 Database 간 DB Connection을 진행한다. |

2. 인터페이스 각 구간의 보안 취약점을 분석한다.

분석된 인터페이스 기능을 대상으로 보안 취약점을 분석한다. 각 단계 영역별로 일어날 수 있는 시나리오를 가정하여 자세하게 분석한다.

<표 2-22> 인터페이스 송수신부의 보안 취약점 분석 예시(인터페이스 객체 전송 예시)

| 단계 | 영역 | 보안 취약점 |
|---|---|---|
| 송신 데이터 선택 | Application | - 송신 데이터 선택 객체 접근 권한 탈취 가능<br>- 객체 선택 시 Sql 검증 필요 |
| | DataBase | - 불완전한 입력값으로 인한 Sql Injection |
| | Network | - DB 정보 전달 시 탈취 가능성, 악의적 서버 변조 가능 |

| 단계 | 영역 | 보안 취약점 |
|---|---|---|
| 송신 객체 생성 | Application | - 생성 객체 접근 권한 탈취 가능<br>- 객체 생성 시 악의적인 코드 삽입 가능 |
| 인터페이스 송신 | Application | - 송신 시점에서 악의적인 사이트로 전달 가능(수신 변조) |
| | Network | - 송수신 네트워크 통신 내역 탈취 및 위변조 |
| 인터페이스 수신 | Application | - 파싱 시 악의적 파서를 만들어 공격 가능<br>- 수신 객체 접근 권한 탈취 가능 |
| 데이터 처리 결과 전송 | Application | - 데이터 처리 객체 접근 권한 탈취 가능<br>- 악의적 데이터 삽입 및 위변조 가능 |
| | DataBase | - 프러시저에 악의적 입력값 삽입 가능(sql injection)<br>- 에러 처리 미흡으로 강제적 내부 오류 발생 시 미인지 가능 |
| | Network | - DB 정보 전달 시 탈취 가능성, 악의적으로 반환값 변조 가능 |

2 분석된 보안 취약점을 근거로 인터페이스 보안 기능을 적용한다.

분석된 인터페이스 기능 및 보안 취약점을 근거로 인터페이스에 보안 기능을 적용한다. 구현 방법에 따라 다양한 단계 및 관점이 있을 수 있으나 일반적으로 크게 네트워크 및 애플리케이션 데이터베이스 영역에서 보안 기능을 적용한다.

1. 네트워크 구간에 보안 기능을 적용한다.

인터페이스 송수신 간 중간자에 의한 데이터 탈취 위변조를 막기 위해서는 네트워크 트래픽에 대한 암호화가 필요하다. 네트워크 구간 암호화를 위해서는 인터페이스 아키텍처에 따라 다양한 방식으로 보안 기능을 적용한다.

<표 2-23> 네트워크 구간 보안 기능 적용 시 고려 사항 예시

| 단계 | 고려 사항 | 보안 기능 적용 |
|---|---|---|
| Transport Layer Network 보안 | 상대방 인증을 적용 | IPSec AH(Authentication Header) 적용, IKE(Internet Key Exchange) 프로토콜 적용 |
| | 데이터 기밀성 보장 필요 | IPSec ESP(Encapsulation Security Payload) 적용 |
| | End-to-End 보안 적용 | IPSec Transport mode 적용 |
| Application Layer Network 보안 | 서버만 공개 키 인증서를 가지고 통신(위험 분산) | SSL(Secure Soket Layer)의 서버 인증 모드 운영 |
| | 연결 단위 외 메시지 단위로도 인증 및 암호화 필요 | S-HTTP 적용하여 메시지 암호화(상호 인증 필요, 성능 일부 저하됨) |