


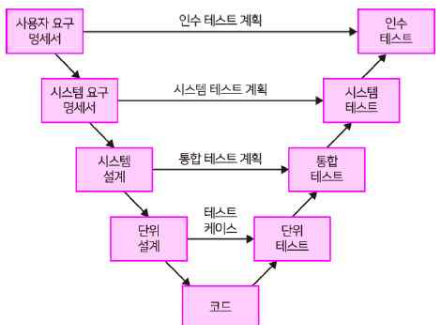
5과목

Chapter 1. 소프트웨어 개발 방법론 활용

090 소프트웨어 생명주기 모델★

• 소프트웨어 생명주기(SDLC; Software Life Cycle): 시스템의 요구분석부터 유지보수까지의 소프트웨어 개발 전체 과정을 체계적으로 정리하여 표준화한 것

• 소프트웨어 생명주기 모델 종류

| 종류 | 설명 |
|---------------------------------|--|
| 폭포수 모델 (Waterfall Model) | <ul style="list-style-type: none"> - 폭포에서 떨어진 물은 거슬러 올라갈 수 없는 것처럼 소프트웨어 개발 시, 각 단계를 마무리지은 후, 다음 단계로 넘어가는 모델 - Bohem이 제시한 고전적 생명주기 모델로, 선형 순차적 모델이라고 함 - 모델을 적용한 경험과 성공 사례가 많으며, 단계별 정의와 산출물이 명확하나, 중간에 요구사항 변경 어려움 - 절차 <div style="border: 1px solid black; padding: 5px; display: inline-block;"> 타당성 검토 → 계획 → 요구사항분석 → 설계 → 구현 → 테스트 → 유지보수 </div> |
| 프로토타입 모델 (Prototyping Model) | <ul style="list-style-type: none"> - 사용자의 요구사항을 실제 개발될 소프트웨어에 대한 프로토타입(Prototype)을 만들어 최종 결과물을 예측하는 모델 - 프로토타입은 구현 단계에서 구현 골격으로 활용 - 절차  |
| V 모델 (V Model) | <ul style="list-style-type: none"> - 폭포수 모델의 확장형으로 생명주기 단계별로 테스트 단계가 추가되어 개발 작업과 검증 작업 사이의 관계를 명확히 들어낸 모델 - Perry에 의해 제안되었으며 세부적인 테스트 과정으로 구성되어 신뢰도 높은 시스템을 개발하는데 효과적임 - 절차: 위에서 아래 방향(↘)으로 진행하다가 개발 단계를 거치면서 아래에서 위로(↗)로 향함  |
| 나선형 모델 (Spiral Model) | <ul style="list-style-type: none"> - 폭포수 모델과 프로토타입 모델의 장점에 위험 분석 기능을 추가하여 점진적으로 완벽한 시스템으로 개발해나가는 모델 - 대형 프로젝트 비교적 적합하나, 프로젝트 관리가 어려움 - 절차 |

| | | |
|--|-------|------------------------|
| | 영역 공학 | 영역 분석, 영역 설계, 핵심 자산 구현 |
| | 응용 공학 | 제품 요구 분석, 제품 설계, 제품 구현 |

092 비용 산정 모델과 일정관리 모델★

• 소프트웨어 비용 산정: 소프트웨어 개발에 소요되는 인원, 자원, 기간 등을 파악하여 실행 가능한 계획을 수립하기 위해 비용을 산정하는 기법이다.

• 비용 산정 기법 종류

- 하향식 비용 산정 기법: 전문가를 통해 비용 산정

| 종류 | 설명 |
|-----------|--|
| 전문가 감정 기법 | - 조직 내 경험이 많은 두 명 이상의 전문가에게 비용 산정을 의뢰하는 기법 - 빠르고 편리하게 비용을 산정할 수 있으나, 개인적, 주관적 산정 |
| 델파이 기법 | - 전문가 감정 기법의 주관적인 판단을 보완하기 위해 많은 전문가의 의견을 종합하여 비용을 산정하는 기법 - 1명의 조정자와 여러 명의 전문가로 구성 |

- 상향식 비용 산정 기법: 요구사항과 기능에 따라 비용 산정

| 종류 | 설명 | | | | | | | | |
|-------------------------------------|--|----|----|-----------------------|---|-------------------------------|---|--------------------------|--|
| LOC 기법 (source Line Of Code) | - 소프트웨어 각 기능의 원시 코드 라인 수의 낙관치, 중간치, 비관치를 측정하여 예측치를 구하고, 이를 이용하여 비용 산정 <div style="border: 1px solid black; padding: 5px; margin: 5px 0;"> $\text{예측치} = \frac{o + 4m + p}{6} \quad (o: \text{낙관치}, m: \text{중간치}, p: \text{비관치})$ <p>낙관치: 가장 적게 측정된 코드 라인수 중간치: 측정된 모든 코드 라인 수의 평균 비관치: 가장 많이 측정된 코드 라인 수</p> </div> - 측정이 쉽고 이해하기 쉬워 많이 사용 - 예측치를 이용하여 생산성, 노력, 개발 기간 등의 비용 산정 | | | | | | | | |
| Man Month | - 한 사람이 1개월 동안 할 수 있는 일의 양을 기준으로 프로젝트 비용을 산정 <div style="border: 1px solid black; padding: 5px; margin: 5px 0;"> $\text{Man Month} = \text{LOC} / \text{개발자 월 생산성}$ $\text{프로젝트 기간} = \text{Man Month} / \text{참여 개발자 수}$ </div> | | | | | | | | |
| COCOMO (Constructive Cost Model) | - Boehm이 제안한 모델로 LOC(원시 코드 라인 수)에 따라 비용 산정 - 비용 산정 결과는 프로젝트를 완성하는 데 필요한 노력(Man-Month)로 표현 - 프로젝트의 복잡도 또는 원시 프로그램의 규모에 따라 조직형, 반분리형, 임베디드형으로 분류 <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>유형</th><th>설명</th></tr> </thead> <tbody> <tr> <td>조직형 (Organic Mode)</td><td>- 기관 내부에서 개발된 중·소 규모의 소프트웨어로 일괄 자료 처리, 과학 기술 계산용, 비즈니스 자료 개발용 - 5만(50KDSI) 라인 이하의 소프트웨어 개발</td></tr> <tr> <td>반 분리형 (Semi-Detached Mode)</td><td>- 조직형과 임베디드형의 중간형으로 트랜잭션 처리 시스템이나 운영체제, 데이터베이스 관리 시스템, 컴파일러 개발용 - 30만(300KDSI) 라인 이하의 소프트웨어 개발</td></tr> <tr> <td>임베디드형 (Embedded Mode)</td><td>- 초대형 규모의 트랜잭션 처리 시스템, 운영체제, 실시간 처리 시스템 등 시스템 프로그램 개발용 - 30만(300KDSI) 라인 이상의 소프트웨어 개발</td></tr> </tbody> </table> | 유형 | 설명 | 조직형 (Organic Mode) | - 기관 내부에서 개발된 중·소 규모의 소프트웨어로 일괄 자료 처리, 과학 기술 계산용, 비즈니스 자료 개발용 - 5만(50KDSI) 라인 이하의 소프트웨어 개발 | 반 분리형 (Semi-Detached Mode) | - 조직형과 임베디드형의 중간형으로 트랜잭션 처리 시스템이나 운영체제, 데이터베이스 관리 시스템, 컴파일러 개발용 - 30만(300KDSI) 라인 이하의 소프트웨어 개발 | 임베디드형 (Embedded Mode) | - 초대형 규모의 트랜잭션 처리 시스템, 운영체제, 실시간 처리 시스템 등 시스템 프로그램 개발용 - 30만(300KDSI) 라인 이상의 소프트웨어 개발 |
| 유형 | 설명 | | | | | | | | |
| 조직형 (Organic Mode) | - 기관 내부에서 개발된 중·소 규모의 소프트웨어로 일괄 자료 처리, 과학 기술 계산용, 비즈니스 자료 개발용 - 5만(50KDSI) 라인 이하의 소프트웨어 개발 | | | | | | | | |
| 반 분리형 (Semi-Detached Mode) | - 조직형과 임베디드형의 중간형으로 트랜잭션 처리 시스템이나 운영체제, 데이터베이스 관리 시스템, 컴파일러 개발용 - 30만(300KDSI) 라인 이하의 소프트웨어 개발 | | | | | | | | |
| 임베디드형 (Embedded Mode) | - 초대형 규모의 트랜잭션 처리 시스템, 운영체제, 실시간 처리 시스템 등 시스템 프로그램 개발용 - 30만(300KDSI) 라인 이상의 소프트웨어 개발 | | | | | | | | |
| Putnam 모형 | - 소프트웨어 개발 주기 단계별 노력의 분포 가정하여 비용 산정 - 푸트남(Putnam)이 제안한 것으로, 생명주기 예측 모형이라고도 함 | | | | | | | | |

| | <ul style="list-style-type: none">- 시간에 따른 함수로 표현되는 Rayleigh-Norden 곡선의 노력 분포도를 기초로 함- 비용 산정 자동화 추정 도구로 SLIM 사용 | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|----------|-----|----|--|-------|----|----|----|-----------------|---|---|---|-------------------|---|---|---|-------------------|---|---|---|--------|---|----|----|-------------------------|---|---|----|
| 기능점수(FP: Function Point) 모형 | <ul style="list-style-type: none">- 요구 기능을 증가시키는 인자별로 가중치를 부여하고 요인별 가중치를 합산하여 총 기능의 점수를 계산하여 비용 산정 | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | <div>기능점수(FP) = 총 기능 점수 × [0.65 + 0.1 × 총 영향도]</div> | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | <ul style="list-style-type: none">- 기능별 가중치 | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | <table><tr><th>소프트웨어 기능</th><th colspan="3">가중치</th></tr><tr><th>증대 요인</th><th>단순</th><th>보통</th><th>복잡</th></tr><tr><td>자료 입력 (입력양식)</td><td>3</td><td>4</td><td>6</td></tr><tr><td>정보 출력 (출력 보고서)</td><td>4</td><td>5</td><td>7</td></tr><tr><td>명령어 (사용자 질의 수)</td><td>3</td><td>4</td><td>5</td></tr><tr><td>데이터 파일</td><td>7</td><td>10</td><td>15</td></tr><tr><td>필요한 외부 루틴과의 인터페이스</td><td>5</td><td>7</td><td>10</td></tr></table> | 소프트웨어 기능 | 가중치 | | | 증대 요인 | 단순 | 보통 | 복잡 | 자료 입력 (입력양식) | 3 | 4 | 6 | 정보 출력 (출력 보고서) | 4 | 5 | 7 | 명령어 (사용자 질의 수) | 3 | 4 | 5 | 데이터 파일 | 7 | 10 | 15 | 필요한 외부 루틴과의 인터페이스 | 5 | 7 | 10 |
| | 소프트웨어 기능 | 가중치 | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | 증대 요인 | 단순 | 보통 | 복잡 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 자료 입력 (입력양식) | 3 | 4 | 6 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 정보 출력 (출력 보고서) | 4 | 5 | 7 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 명령어 (사용자 질의 수) | 3 | 4 | 5 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 데이터 파일 | 7 | 10 | 15 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 필요한 외부 루틴과의 인터페이스 | 5 | 7 | 10 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| <ul style="list-style-type: none">- 비용 산정 자동화 도구로 ESTIMACS 사용 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

- 일정관리 모델: 프로젝트가 일정 기한 내에 적절하게 완료될 수 있도록 관리하는 모델

| 모델 종류 | 설명 | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|------|----|--|--|-----------|--|-----|----|--|--|--|--|--|----------|----|--|--|--|--|--|-----------|----|--|--|--|--|--|-------|
| CMP 기법 (Critical Path Method) | <div><div><div><div><div>시작</div><div>2일</div><div>A</div><div>2일</div><div>B</div><div>3일</div><div>C</div><div>3일</div><div>완료</div></div><div><div>3일</div><div>D</div><div>5일</div><div>F</div><div>4일</div></div><div><div>3일</div><div>E</div><div>5일</div><div>F</div></div><div><div>5일</div><div>D</div><div>5일</div><div>F</div></div></div></div><div><div>시작→A→B→C→완료: 2 + 2 + 3 + 3 = 10일</div><div>시작→A→E→F→완료: 2 + 3 + 5 + 4 = 14일</div><div>시작→D→F→완료: 3 + 5 + 4 = 12일</div><div>따라서, 임계 경로는 14일이다.</div></div></div> | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| PERT (Program Evaluation and Review Technique) | <div><div><div>일의 순서를 계획적으로 정리하기 위한 기법으로 낙관치, 중간치, 비관치를 사용하여 일정을 관리하는 기법</div><div>작업 예측치 = (낙관치 + (4 × 기대치) + 비관치) / 6</div><div>과거에 경험이 없어서 소요 기간 예측이 어려운 소프트웨어에 사용</div></div><div><div>프로젝트의 시작과 끝을 그래픽으로 표시, 각 작업 일정을 수평 막대(Bar) 형태로 표현한 차트로 시간선(Time-Line) 차트라고도 함</div><div>수평 막대의 길이는 작업에 필요한 시간을 나타냄</div></div></div> | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 간트 차트 (Gantt Chart) | <table><tr><th>작업단계</th><th colspan="5">일정</th><th>산출물</th></tr><tr><td>계획</td><td></td><td></td><td></td><td></td><td></td><td>프로젝트 정의서</td></tr><tr><td>분석</td><td></td><td></td><td></td><td></td><td></td><td>요구 분석 명세서</td></tr><tr><td>구현</td><td></td><td></td><td></td><td></td><td></td><td>원시 코드</td></tr></table> | 작업단계 | 일정 | | | | | 산출물 | 계획 | | | | | | 프로젝트 정의서 | 분석 | | | | | | 요구 분석 명세서 | 구현 | | | | | | 원시 코드 |
| 작업단계 | 일정 | | | | | 산출물 | | | | | | | | | | | | | | | | | | | | | | | |
| 계획 | | | | | | 프로젝트 정의서 | | | | | | | | | | | | | | | | | | | | | | | |
| 분석 | | | | | | 요구 분석 명세서 | | | | | | | | | | | | | | | | | | | | | | | |
| 구현 | | | | | | 원시 코드 | | | | | | | | | | | | | | | | | | | | | | | |

093 소프트웨어 개발 표준★

• 소프트웨어 개발 표준: 소프트웨어 개발 단계에서 수행하는 품질 관리에 사용되는 국제적 표준 (ISO/IEC 12207, CMMI, SPICE)

| 프로세스 구분 | 내용 |
|--------------|---|
| 기본 생명주기 프로세스 | 획득, 공급, 개발, 운영, 유지보수 |
| 지원 생명주기 프로세스 | 품질 보증, 검증, 확인, 활동 검토, 감사, 문서화, 형상관리, 문제해결 |
| 조직 생명주기 프로세스 | 기반 구조, 관리, 개선, 훈련 |

• CMMI(능력 성숙도 통합 모델, Capability Maturity Model Intergration): 소프트웨어 개발 조직의 업무 능력 및 조직의 성숙도를 평가하는 모델

| 수준 | 단계 | 설명 |
|----|--|--|
| 1 | 초기화 단계 (Initial) | - 정의된 프로세스 없음, 예측 불가 - 작업자의 능력에 따라 성공 여부 |
| 2 | 관리 단계 (Managed) | - 특정한 프로젝트 내의 프로세스 정의 및 수행 - 프로젝트 관리 시스템 생성 |
| 3 | 정의 단계 (Defined) | - 조직의 표준 프로세스에 따라 업무 수행 |
| 4 | 정량적 관리 단계 (Quantitatively Managed) | - 정량적 기법을 활용하여 프로세스 통제 - 프로젝트 정량적 관리 및 통제 |
| 5 | 최적화 단계 (Optimizing) | 프로세스 역량 향상을 위해 지속적 개선, 내재화 |

- SPICE(Software Process Imporvement and Capability dEtermination): 소프트웨어 처리 개선 및 능력 평가 기준에 대한 국제 표준

| 수준 | 단계 | 설명 |
|----|--------|---|
| 0 | 불안정 단계 | 프로세스가 구현되지 않았거나, 프로세스가 목적을 달성하지 못한 단계 |
| 1 | 수행 단계 | 프로세스가 수행되고, 목적을 달성한 단계 |
| 2 | 관리 단계 | 정의된 자원의 한도 내에서 프로세스가 작업 산출물을 인도하는 단계 |
| 3 | 확립 단계 | 소프트웨어 공학 원칙에 기반하여 정의된 프로세스 수행되는 단계 |
| 4 | 예측 단계 | 프로세스가 목적 달성을 위해 통제되고, 양적인 측정을 통해 일관되게 수행되는 단계 |
| 5 | 최적화 단계 | 프로세스 수행을 최적화하고, 지속적인 개선을 통해 업무 목적을 만족하는 단계 |

094 테일러링 기준★

• 테일러링(Tailoring): 조직의 프로젝트 상황, 목적, 특성에 맞게 표준 프로세스를 커스터마이징하는 작업

- 테일러링 프로세스

| 프로세스 | 설명 |
|-----------------|--|
| 프로젝트 특징 정의 | 프로젝트의 특징을 파악하고, 목적, 현 상황, 프로젝트에 대한 이해를 하는 단계 |
| 표준 프로세스 선정 및 검증 | 표준 프로세스 선정, 검증 |
| 상위 수준의 커스터마이징 | 비즈니스 특성에 맞는 생명주기 정의, 개발 단계 조정 |
| 세부 커스터마이징 | WBS(Work Breakdown Structure) 적용, 일정 수립 |
| 테일러링 문서화 | 테일러링 내용 문서화, 검토 및 승인 |

- 테일러링 기준

| 구분 | 기준 | 설명 |
|--------|-------|--|
| 내부적 기준 | 목표 환경 | 시스템 개발 유형 및 기술 환경이 서로 다른 경우, 테일러링 필요 |
| | 요구사항 | 소프트웨어 생명주기 활동에서 개발/운영/유지보수 등 프로젝트 상 요구 |

| | | |
|--------|-------------|---|
| | 프로젝트 규모 | 사항 우선 순위가 서로 다른 경우 테일러링 필요 |
| | 보유 기술 | 비용, 인력, 기간과 같은 프로젝트 특성이 서로 다른 경우 테일러링 필요 |
| | | 구성원 역량, 산출물, 개발방법론 등이 서로 다른 경우 테일러링 필요 |
| 외부적 기준 | 법적 제약 | 프로젝트 대상 도메인에 따라 적용되는 컴플라이언스(Compliance)가 서로 다른 경우 테일러링 필요 |
| | 국제 표준 품질 기준 | 금융, 제도, 분야별 표준 품질 기준이 다른 경우 테일러링 필요 |

095 소프트웨어 개발 프레임워크★

- 소프트웨어 개발 프레임워크: 소프트웨어 개발에 공통적으로 사용되는 구성요소와 아키텍처를 손쉽게 구현할 수 있도록 여러 가지 기능을 제공해주는 반제품 상태의 개발 소프트웨어
- 소프트웨어 개발 프레임워크의 특징

| |
|---|
| <ul style="list-style-type: none"> - 소프트웨어 디자인 패턴을 반제품 소프트웨어 상태로 집적화시킨 것 - 도메인별로 필요한 서비스 컴포넌트를 사용하여 재사용성 확대와 성능 보장 - 프레임워크의 동작 원리를 그 제어 흐름의 일반적인 프로그램 흐름과 반대로 동작한다고 해서 IoC(Inversion of Control)라고 불림 - 개발해야 할 애플리케이션의 일부분이 이미 구현되어 있어 동일한 로직 반복 감소 |
|---|

- 소프트웨어 개발 프레임워크 기대효과

| |
|--|
| <ul style="list-style-type: none"> - 개발할 소프트웨어에 대한 품질보증 - 소프트웨어 개발 용이성 증가 - 소프트웨어 개발 변경 발생 시, 변경 용이 - 공통 컴포넌트 재사용으로 중복 예산 절감 - 표준화된 연계모듈 활용으로 상호 운용성 향상 - 개발표준에 의한 모듈화로 유지보수 용이 - 시스템의 복잡도 감소 - 생산성 향상과 유지보수성 향상 |
|--|

Chapter 2. IT프로젝트 정보시스템 구축관리

096 네트워크 관련 신기술★★★

• 네트워크 관련 신기술

| 기술 | 설명 |
|---|--|
| 소프트웨어 정의 네트워크 (SDN: Software Defined Networking) | <ul style="list-style-type: none"> - 네트워크를 제어부, 데이터 전달부로 분리하여 네트워크 관리자보다 효율적으로 네트워크를 제어, 관리할 수 있는 기술 - 기존의 라우터, 스위치 등과 같이 하드웨어에 의존하는 네트워크 체계에서 안정성, 속도, 보안 등을 소프트웨어로 제어 관리하기 위해 개발됨 - 네트워크 장비의 펌웨어 업그레이드를 통해 사용자의 직접적인 데이터 전송 경로 관리가 가능하고, 기존 네트워크에는 영향을 주지 않으면서 특정 서비스의 전송 경로 수정을 통하여 인터넷상에서 발생하는 문제를 처리할 수 있음 |
| 소프트웨어 정의 데이터센터(SDDC :Software Defined Data Center) | <ul style="list-style-type: none"> - 컴퓨팅, 네트워킹, 스토리지, 관리 등을 모두 소프트웨어로 정의 함 - 인력 개입 없이 소프트웨어 조작만으로 자동 제어 관리 - 데이터센터 내 모든 자원을 가상화하여 서비스 |
| 소프트웨어 정의 스토리지 (SDS: Software Defined Storage) | <ul style="list-style-type: none"> - 가상화를 적용하여 필요한 공간만큼 나눠 사용할 수 있도록 하며 서버 가상화와 유사 - 컴퓨팅 소프트웨어로 규정하는 데이터 스토리지 체계이며, 일정 조직 내 여러 스토리지를 하나처럼 관리하고 운용하는 컴퓨터 이용환경 - 스토리지 자원을 효율적으로 나누어 쓰는 방법 |
| 메시 네트워크 (Mesh Network) | <ul style="list-style-type: none"> - 다른 국을 향하는 호출이 중계에 의하지 않고 직접 접속되는 그물 모양의 네트워크 - 통신량이 많은 비교적 소수의 국 사이에 구성될 경우 경제적이며 간편하지만, 다수의 국 사이에는 회선이 세분화되어 비경제적임 - 해당 형태의 무선 네트워크의 경우 대용량을 빠르고 안전하게 전달할 수 있어 행사 장이나 군 등에서 활용 |
| 피코넷 (PICONET) | 여러 개의 독립된 통신장치가 UWB(Ultra Wideband) 기술 또는 블루투스 기술을 사용하여 통신망을 형성하는 무선 네트워크 기술 |
| UWB (Ultra Wide Band) | <ul style="list-style-type: none"> - 매우 낮은 전력을 사용하여, 초광대역 주파수 대역으로 짧은 거리에서 많은 양의 디지털 데이터를 전송하는 무선 전송 기술 - 땅 속, 벽을 통과하여 신호 전송이 가능하여, 재해 상황에서 인명 구조 활용 |
| 근거리 무선 통신 (NFC: Near Field Communication) | <ul style="list-style-type: none"> - 고주파를 이용한 근거리 무선 통신 기술로, 10cm 이내에서 저전력, 비접촉, 양방향 무선 통신 기술 - 모바일 기기를 활용한 결제, 교통, 출입 통제 등에 사용 |
| SON (Self Organizing Network) | 노드 간 상호작용으로 스스로 망을 구성, 최적화하는 자율적 네트워크 기술 |
| 사물 인터넷 (IoT: Internet of Things) | 정보 통신 기술을 기반으로 다양한 사물에 센서와 통신 기능을 내장하여 무선 통신을 통해 사물을 인터넷에 연결하는 기술 |
| 클라우드 기반 HSM (Cloud-based Hardware Security Module) | <ul style="list-style-type: none"> - 클라우드(데이터센터) 기반 암호화 키 생성, 처리, 저장 등을 하는 보안 기술 - 클라우드에 인증서를 저장하므로 기존 HSM 기기나 휴대폰에 인증서를 저장해 다닐 필요가 없음 |
| 파스-타 (PaaS-TA) | 국내 IT 서비스 경쟁력 강화를 목표로 개발되었으며 인프라 제어 및 관리 환경, 실행 환경, 개발 환경, 서비스 환경, 운영환경으로 구성되어 있는 개방형 클라우드 컴퓨팅 플랫폼 |
| 스마트 그리드 (Smart Grid) | 전기 및 정보통신기술을 활용하여 전력망을 지능화, 고도화함으로써 고품질의 전력서비스를 제공하고 에너지 이용효율을 극대화하는 전력망 |
| MQTT (Message Queuing Telemetry Transport) | TCP/IP 기반 네트워크에서 동작하는 발행-구독 기반의 메시징 프로토콜로 최근 IoT 환경에서 자주 사용되고 있는 프로토콜 |

| | |
|--|---|
| Zing | 기기를 키오스크에 갖다 대면 원하는 데이터를 바로 가져올 수 있는 기술로 10cm 이내 근접 거리에서 기가급 속도로 데이터 전송이 가능한 초고속 근접무선통신 (NFC : Near Field Communication) 기술 |
| SSO (Single Sign On) | 시스템이 몇 대가 되어도 하나의 시스템에서 인증에 성공하면 다른 시스템에 대한 접근 권한도 얻는 시스템 |
| Wi-SUN (Wireless Smart Utility Network) | 스마트 그리드와 연계하여 전기, 수도, 가스 공급자와 사용자가 자체 가망 구축 형태의 비면허대역 무선 네트워크를 이용하여 관리할 수 있게 하는 무선 통신 기술 |

097 네트워크 장비★

• 네트워크 설치 구조(Topology): 통신망을 구성하는 장치들을 배치하는 방법

| 구조 | 설명 |
|--------------|--|
| 성(Star)형 구조 | <ul style="list-style-type: none"> - 각 단말장치가 중앙 허브에 포인트 투 포인트(Point-to-Point) 방식으로 연결 - 소규모 네트워크 설치, 네트워크 재구성(단말의 추가/제거) 쉬움 - 중앙 허브가 고장나면 전체 네트워크 정지, 하나의 단말장치가 고장나는 경우, 다른 단말장치에 영향을 주지 않음 |
| 링(Ring)형 구조 | <ul style="list-style-type: none"> - 모든 단말장치가 하나의 링에 순차적으로, 포인트 투 포인트(Point-to-Point) 방식으로 연결된 형태 - 데이터는 단방향, 양방향 전송 모두 가능하나, 단방향 링의 경우, 단말장치가 하나라도 고장나면 전체 통신망 정지 |
| 버스(Bus)형 구조 | <ul style="list-style-type: none"> - 하나의 네트워크에 여러 대의 단말장치가 연결된 형태 - 단말장치가 고장나더라도 네트워크 전체에 영향을 주지 않음 |
| 트리(Tree)형 구조 | <ul style="list-style-type: none"> - 각 단말장치가 계층적으로 연결되어 있는 구성 - 분산처리 시스템을 구성하는 방식 |

• 네트워크 장비: 서로 통신을 할 수 있는 컴퓨터, 스위치, 라우터, 광전송 장비

| 장비 종류 | 설명 | | | | | | | | |
|--|---|------|----|--|--|-----------------------------------|---|----------------------------------|--|
| 스위치 장비 | <ul style="list-style-type: none"> - LAN과 LAN을 연결하여 더 큰 LAN을 만드는 장치 - OSI 7계층의 2계층에서 사용 | | | | | | | | |
| 라우터 장비 | <ul style="list-style-type: none"> - 스위치를 서로 연결하여 송수신 전송 경로중 최적화된 경로를 설정하고, 설정된 경로를 따라 트래픽을 전달하는 장치 - OSI 7계층의 3계층에서 사용 <table border="1"> <thead> <tr> <th>프로토콜</th><th>설명</th></tr> </thead> <tbody> <tr> <td>RIP (Routing Information Protocol)</td><td> <ul style="list-style-type: none"> - 거리 벡터 라우팅 프로토콜 - 소규모 네트워크 환경에 적합 - 최대 홉 카운트를 15홉 이하 제한 - 최단경로탐색 시, Bellman-Ford 알고리즘 사용 </td></tr> <tr> <td>OSPF (Open Shorter Path First)</td><td> <ul style="list-style-type: none"> - 대규모 네트워크 환경에 적합 - 홉 카운트에 제한 없음 - 최단경로탐색 시, Dijkstra 알고리즘 사용 </td></tr> <tr> <td>BGP (Border Gateway Protocol)</td><td> <ul style="list-style-type: none"> - 자율 시스템(AS)간 라우팅에 경로 정보를 교환하기 위한 라우팅 프로토콜 </td></tr> </tbody> </table> | 프로토콜 | 설명 | RIP (Routing Information Protocol) | <ul style="list-style-type: none"> - 거리 벡터 라우팅 프로토콜 - 소규모 네트워크 환경에 적합 - 최대 홉 카운트를 15홉 이하 제한 - 최단경로탐색 시, Bellman-Ford 알고리즘 사용 | OSPF (Open Shorter Path First) | <ul style="list-style-type: none"> - 대규모 네트워크 환경에 적합 - 홉 카운트에 제한 없음 - 최단경로탐색 시, Dijkstra 알고리즘 사용 | BGP (Border Gateway Protocol) | <ul style="list-style-type: none"> - 자율 시스템(AS)간 라우팅에 경로 정보를 교환하기 위한 라우팅 프로토콜 |
| 프로토콜 | 설명 | | | | | | | | |
| RIP (Routing Information Protocol) | <ul style="list-style-type: none"> - 거리 벡터 라우팅 프로토콜 - 소규모 네트워크 환경에 적합 - 최대 홉 카운트를 15홉 이하 제한 - 최단경로탐색 시, Bellman-Ford 알고리즘 사용 | | | | | | | | |
| OSPF (Open Shorter Path First) | <ul style="list-style-type: none"> - 대규모 네트워크 환경에 적합 - 홉 카운트에 제한 없음 - 최단경로탐색 시, Dijkstra 알고리즘 사용 | | | | | | | | |
| BGP (Border Gateway Protocol) | <ul style="list-style-type: none"> - 자율 시스템(AS)간 라우팅에 경로 정보를 교환하기 위한 라우팅 프로토콜 | | | | | | | | |
| 광전송 장비 | <p>광케이블을 이용하여 스위칭 노드를 묶어주는 시스템</p> <table border="1"> <thead> <tr> <th>기술</th><th>설명</th></tr> </thead> <tbody> <tr> <td>SONET (Synchronous Optical Network)</td><td> <ul style="list-style-type: none"> - 고속 디지털 통신을 위한 광전송 시스템 표준 규격 </td></tr> <tr> <td>WDM</td><td> <ul style="list-style-type: none"> - 광섬유를 이용한 통신 기술 </td></tr> </tbody> </table> | 기술 | 설명 | SONET (Synchronous Optical Network) | <ul style="list-style-type: none"> - 고속 디지털 통신을 위한 광전송 시스템 표준 규격 | WDM | <ul style="list-style-type: none"> - 광섬유를 이용한 통신 기술 | | |
| 기술 | 설명 | | | | | | | | |
| SONET (Synchronous Optical Network) | <ul style="list-style-type: none"> - 고속 디지털 통신을 위한 광전송 시스템 표준 규격 | | | | | | | | |
| WDM | <ul style="list-style-type: none"> - 광섬유를 이용한 통신 기술 | | | | | | | | |

| | |
|--|---|
| | (Wavelength Division Multiplexing) <ul style="list-style-type: none"> - 파장이 서로 다른 복수의 광신호를 동시에 이용하는 것으로, 광섬유를 다중화하는 방식 - 빛의 파장 축과 파장이 다른 광선은 서로 간섭을 일으키지 않는 성질 이용 |
|--|---|

098 소프트웨어 관련 신기술★★★

• 소프트웨어 관련 신기술

| 기술 | 설명 |
|---------------------------------------|--|
| 인공지능 (AI: Artificial Intelligence) | <ul style="list-style-type: none"> - 인간의 두뇌와 같이 컴퓨터가 인간의 지능적 작업을 수행하는 시스템 - 신경망, 자연어 처리, 컴퓨터 비전 등에서 응용 |
| 기계학습 (Machine Learning) | <ul style="list-style-type: none"> - 인간이 학습을 하듯 컴퓨터에 데이터를 입력하여 학습시키고, 답을 예측하게 만드는 것 - 알고리즘 개발이 어려운 문제의 해결에 유용하며, 학습 문제에 따라 지도학습, 비지도학습, 강화학습으로 나누어짐 |
| 텐서플로 (TensorFlow) | 구글 브레인 팀이 제작하여 공개한 기계 학습을 위한 오픈소스 소프트웨어 라이브러리 |
| 증강현실 (AR: Augmented Reality) | 실제와 유사하지만 실체가 아닌 환경이나 상황을 구현하는 기술 |
| 그레이웨어 (Grayware) | <ul style="list-style-type: none"> - 바이러스인지, 평범한 소프트웨어인지 구분하기 어려운 프로그램 - 사용자가 원하지 않는 애드웨어, 트랙웨어, 기타 악성코드 등 |
| 매시업 (Mashup) | 웹에서 제공하는 정보 및 서비스를 융합하여 새로운 소프트웨어, 서비스, 데이터베이스 등을 만드는 기술 |
| 디지털 트윈 | 물리적인 사물과 컴퓨터에 동일하게 표현되는 가상의 모델로 실제 물리적인 자산 대신 소프트웨어로 가상화함으로써 실제 자산의 특성에 대한 정확한 정보를 얻을 수 있고, 자산 최적화, 돌발사고 최소화, 생산성 증가 등 설계부터 제조, 서비스에 이르는 모든 과정의 효율성을 향상 시킬 수 있는 모델 |
| Baas | <ul style="list-style-type: none"> - 블록체인(Blockchain) 개발환경을 클라우드로 서비스하는 개념 - 블록체인 네트워크에 노드의 추가 및 제거가 용이 - 블록체인의 기본 인프라를 추상화하여 블록체인 응용프로그램을 만들 수 있는 클라우드 컴퓨팅 플랫폼 |

099 소프트웨어 개발 보안 정책

• 소프트웨어 개발 보안 관련 법규

| 관련 법규 | 내용 |
|----------|---|
| 개인정보 보호법 | 개인정보 처리 및 보호에 관한 사항 규정 |
| 정보통신망법 | 정보통신망을 통하여 수집, 처리, 보관, 이용되는 개인정보의 보호에 관한 규정 |
| 신용정보법 | 개인신용정보의 취급 단계별 보호조치 및 의무사항에 관한 규정 |
| 위치정보법 | 개인 위치정보 수집, 이용, 제공 파기, 및 정보 주체 권리 규정 |

• Secure SDLC(Software Development Life Cycle)모델 및 방법론

| 구분 | 설명 |
|--|---|
| OWASP CLASP | 활동중심, 역할기반의 프로세스로 구성된 보안 프레임워크로 기존에 운영중인 시스템에 적용하기 쉬움 |
| Open SAMM (Software Assurance Maturity Model) | OWASP에서 개발한 개방형 보안 프레임워크로 점진적 확대 가능 |

| | |
|--|---|
| MS SDL (Security Development Lifecycle) | 마이크로소프트사(MS)가 자사의 소프트웨어 개발에 의무적으로 적용하도록 고안한 보안 강화 프레임워크 |
| Seven TouchPoints | 실무적으로 검증된 개발 보안 방법론 중 하나로써 SW보안의 모범 사례를 SDLC(Software Development Life Cycle)에 통합한 소프트웨어 개발 보안 생명주기 방법론 |

100 하드웨어 관련 신기술★★★

• 하드웨어 관련 신기술

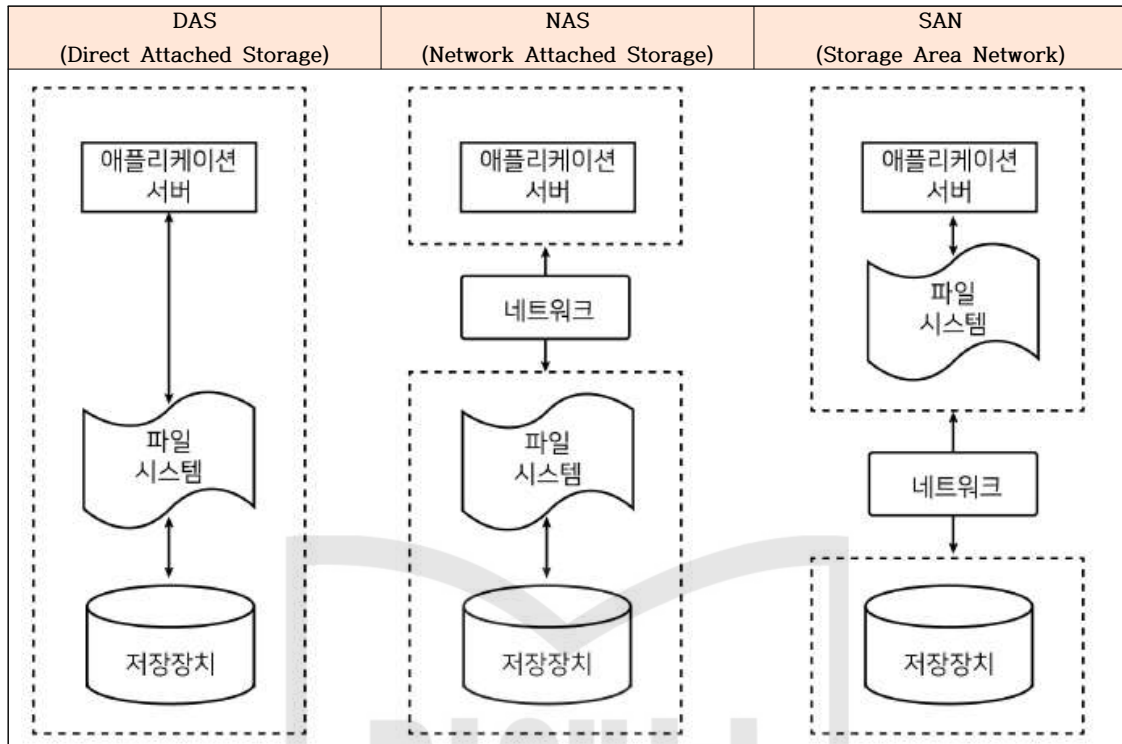
| 기술 | 설명 |
|--|---|
| 클라우드 컴퓨팅 (Cloud Computing) | 인터넷을 통해 가상화된 컴퓨터 시스템 자원을 제공하고, 정보를 클라우드에 연결된 컴퓨터로 처리하는 기술 |
| 도커 (Docker) | - 컨테이너 응용프로그램의 배포를 자동화하는 오픈소스 엔진 - 소프트웨어 컨테이너 안에 응용프로그램들을 배치시키는 일을 자동화해 주는 오픈소스 프로젝트이자 소프트웨어 |
| 쿠버네티스 (Kubernetes) | 리눅스 재단에 의해 관리되는 컨테이너화된 애플리케이션의 배포 자동화, 스케일링을 제공하는 오픈소스 기반의 관리 시스템 |
| 고가용성 솔루션 (HACMP: High Availability Clustering Multi Processing) | 두 개 이상의 시스템을 클러스터로 구성하여, 하나의 시스템에 장애 발생 시, 즉시 다른 시스템으로 대체 작동(Fail Over)하는 기술 |
| N-Screen | - N개의 서로 다른 단말기에서 동일한 콘텐츠를 자유롭게 이용할 수 있는 기술 - PC, TV, 휴대폰에서 원하는 콘텐츠를 끊임없이 자유롭게 이용할 수 있는 서비스 |

101 하드웨어 장비 운영★

- 저장장치(스토리지 시스템): 정보 시스템 구축 및 운영을 위해서 대용량 데이터를 저장하기 위한 장치
- 저장장치의 종류

| 저장장치 | 설명 |
|-----------------------------------|---|
| DAS (Direct Attached Storage) | - 하드디스크와 같은 데이터 저장장치를 호스트 버스 어댑터에 직접 연결하는 방식 - 저장장치와 호스트 기기 사이에 네트워크 디바이스 없이 직접 연결하는 방식으로 구성 |
| NAS (Network Attached Storage) | 서버와 저장장치를 네트워크로 연결하는 방식으로 구성된 스토리지 시스템 |
| SAN (Storage Area Network) | - 각기 다른 운영체제를 가진 여러 기종이 네트워크상에서 동일 저장장치의 데이터를 공유하게 함으로써, 여러 개의 저장장치나 백업 장비를 단일화시킨 시스템 - 네트워크상에 광 채널 스위치의 이점인 고속 전송과 장거리 연결 및 멀티 프로토콜 기능 활용 |

- 저장장치 구성도



102 데이터베이스 관련 신기술 및 데이터베이스 관리 기능★★

• 데이터베이스 관련 신기술

| 기술 | 설명 |
|------------------------------------|---|
| 빅데이터 (Big Data) | <ul style="list-style-type: none"> - 기존의 관리 방법이나 분석 체계로 처리하기 어려운 많은 양의 데이터 - 빅데이터의 특성으로 데이터의 양(Volume), 데이터의 다양성(Variety), 데이터의 속도(Velocity)을 3V라 함 |
| 데이터 웨어하우스 (DW; Data Warehouse) | 사용자의 의사결정에 도움을 주기 위하여 기간 시스템의 데이터베이스에 축적된 데이터를 공통의 형식으로 변환해서 관리하는 데이터베이스 |
| 하둡 (Hadoop) | <ul style="list-style-type: none"> - 오픈 소스를 기반으로 한 분산 컴퓨팅 플랫폼 - 일반 PC급 컴퓨터들로 가상화된 대형 스토리지 형성 - 거대한 데이터 세트를 병렬로 처리할 수 있도록 개발된 자바 소프트웨어 프레임워크로 구글, 야후 등에서 적용 |
| 맵리듀스 (MapReduce) | <ul style="list-style-type: none"> - 구글에서 대용량 데이터를 분산 병렬 컴퓨팅에서 처리하기 위한 목적으로 제작 - 연관성 있는 데이터를 묶어 쪼개는 Map, 중복된 데이터 제거 및 추출을 하는 Reduce 작업을 함 |
| 하이프 (Hive) | 하둡 기반의 데이터 웨어하우스 솔루션으로 SQL과 유사한 HiveQL이라는 쿼리 제공 |
| 타조 (Tajo) | <ul style="list-style-type: none"> - 하둡 기반의 분산 데이터 웨어하우스 프로젝트로 우리나라에서 주도하여 개발 - 맵리듀스를 사용하지 않고, SQL을 사용하여 하둡 분산 파일 시스템 파일을 읽음 |
| 데이터 마이닝 (Data Mining) | 데이터 웨어하우스에 저장된 데이터 집합에서 사용자의 요구에 따라 유용하고 가능성 있는 정보를 도출하는 기법 |
| OLAP(Online Analytical Processing) | 다차원으로 이루어진 데이터로부터 통계적인 요약 정보를 분석, 의사 결정에 활용 |
| 스크래파이 (Scrapy) | 웹 사이트를 크롤링하여 구조화된 데이터를 수집하는 파이썬(Python) 기반의 애플리케이션 프레임워크 |

- 데이터베이스 보안: 외부, 내부에서 DB에 저장된 기밀 정보에 불법적으로 접근하는 것을 막는 행위
- 데이터베이스 보안 3대 요소

| 요소 | 설명 |
|--------------------------|--|
| 기밀성 (Confidentiality) | 접근 체계를 만들어 허가 받지 않은 개인, 시스템의 접근 차단 |
| 무결성 (Integrity) | 절차를 따르지 않고 데이터가 변경될 수 없으며, 데이터의 정확성 및 완전함이 훼손되지 않음을 보장 |
| 가용성 (Availability) | 권한을 가진 개인, 시스템이 원하는 데이터에 대한 원활한 접근 제공 보장 |



Chapter 3. 시스템 보안 구축

103 SW 개발 보안 3요소★★

| 요소 | 설명 |
|--------------------------|--|
| 기밀성 (Confidentiality) | 인가된 사용자에게 대해서만 자원 접근이 가능해야 하는 특성 |
| 무결성 (Integrity) | 인가된 사용자에게 대해서만 자원 수정이 가능하며, 전송 중인 정보는 수정되지 않아야 하는 특성 |
| 가용성 (Availability) | 인가된 사용자는 가지고 있는 권한 범위 내에서 언제든지 자원 접근이 가능해야 하는 특성 |

외우기 Tip! **기밀성, 무결성, 가용성** → **기, 무, 가**(정보보안을 위해 **기**문사들이 **가**다.)

104 Secure SDLC와 Secure Coding★★

• Secure SDLC(Software Development Life Cycle)

- 보안상 안전한 소프트웨어를 개발하기 위해 소프트웨어 개발 생명주기(SDLC; Software Development Life Cycle)에 **보안 강화**를 위한 프로세스
- CLASP, SDL, **Seven Touchpoints** 등이 있음

• 시큐어 코딩(Secure Coding)

- 소프트웨어의 구현 단계에서 발생할 수 있는 보안 취약점들을 최소화하기 위해 보안 요소들을 고려하여 코딩하는 것

• 보안 운영체제(Secure OS)

- 컴퓨터 운영체제의 커널에 보안 기능을 추가한 것으로 운영체제의 보안상 결함으로 인하여 발생 가능한 각종 해킹으로부터 시스템을 보호하기 위하여 사용되는 것

105 보안 점검 항목★★★★

• 입력 데이터 검증 및 표현

| 보안 취약점 | 설명 | 대책 |
|--|--|---|
| SQL 삽입 (SQL Injection) | 웹 응용 프로그램에 SQL을 삽입하여 내부 데이터베이스(DB) 서버의 데이터를 유출 및 변조하고, 관리자 인증을 우회하는 보안 취약점 | 동적 쿼리에 사용되는 입력 데이터에 예약어 및 특수문자가 입력되지 않게 필터링 되도록 설정하여 방지 |
| 경로 조작 및 자원 삽입 | 데이터 입출력 경로를 조작하여 서버 자원을 수정·삭제할 수 있는 보안 취약점 | 사용자 입력값을 식별자로 사용하는 경우, 경로 순회 공격을 막는 필터를 사용하여 방지 |
| 크로스사이트 스크립팅 (XSS: Cross Site Scripting) | 웹페이지에 악의적인 스크립트를 삽입하여 방문자들의 정보를 탈취하거나, 비정상적인 기능 수행을 유발하는 보안 취약점 | - HTML 태그 사용 금지(특히, < 문자 사용 시 <로 변환처리 등) - 특수문자 등록을 방지하기 위해 특수 문자 필터링 |
| 메모리 버퍼 오버플로 | 연속된 메모리 공간을 사용하는 프로그램에서 할당된 메모리의 범위를 넘어선 위치에서 자료를 읽거나 쓰려고 할 때 발생하는 보안 취약점 | 메모리 버퍼를 사용할 경우 적절한 버퍼의 크기를 설정하고, 설정된 범위의 메모리 내에서 올바르게 읽거나 쓸 수 있도록 함으로써 방지 |
| 운영체제 명령어 삽입 | 외부 입력값을 통해 시스템 명령어의 실행을 유도함으로써 권한을 탈취하거나 시스템 장애를 유발하는 보안 취약점 | 웹 인터페이스를 통해 시스템 명령어가 전달되지 않도록 하고, 외부 입력값을 검증 없이 내부 명령어로 사용하지 않음으로써 방지 |

| | | |
|--|---|---|
| 사이트 간 요청 위조 (CSRF: Cross-Site Request Forgery) | 사용자가 자신의 의지와는 무관하게 공격자가 의도한 행위를 특정 웹사이트에 요청하게 하는 보안 취약점 | <ul style="list-style-type: none"> - 입력화면 폼을 작성 시 GET 방식보다 POST 방식 사용 - 입력 폼과 입력처리 프로그램에서 세션별 CSRF 토큰을 사용하여 점검 - 중요 기능의 경우 재인증을 통해 안전하게 실제 요청 여부를 확인하도록 구현 |
| 위험한 형식 파일 업로드 | 악의적인 명령어가 포함된 스크립트 파일을 업로드함으로써 시스템에 손상을 주거나, 시스템을 제어할 수 있는 보안 취약점 | 업로드되는 파일의 확장자 제한, 파일명의 암호화, 웹사이트와 파일 서버의 경로 분리, 실행 속성을 제거하는 등의 방법으로 방지 |
| 신뢰되지 않은 URL 주소로 자동접속 연결 | 입력 값으로 사이트 주소를 받는 경우 이를 조작하여 방문자를 피싱 사이트로 유도하는 보안 취약점 | 연결되는 외부 사이트의 주소를 화이트 리스트로 관리함으로써 방지 |

• 보안 기능

| 보안 취약점 | 설명 | 대책 |
|----------------------|---|---|
| 적절한 인증 없이 중요기능 허용 | 보안검사를 우회하여 인증과정 없이 중요정보 또는 기능에 접근 및 변경 가능한 보안 취약점 | 중요정보나 기능을 수행하는 페이지에서는 재인증 기능을 통해 방지 |
| 부적절한 인가 | 접근제어 기능이 없는 실행경로를 통해 정보 또는 권한 탈취가 가능한 보안 취약점 | 모든 실행경로에 대해 접근제어 검사를 수행하고, 사용자에게는 반드시 필요한 접근권한만 부여하여 방지 |
| 중요한 자원에 대한 잘못된 권한 설정 | 권한 설정이 잘못된 자원에 접근하여 해당 자원을 임의로 사용 가능한 보안 취약점 | 소프트웨어 관리자만 자원을 읽고 쓸 수 있도록 설정하고, 인가되지 않은 사용자의 중요 자원에 대한 접근 여부를 검사함으로써 방지 |
| 취약한 암호화 알고리즘 사용 | 암호화된 환경설정 파일을 해독하여 비밀번호 등의 중요정보를 탈취할 수 있는 보안 취약점 | 안전한 암호화 알고리즘, 안정성이 인증된 암호 모듈을 이용하여 방지 |
| 중요정보 평문 저장 및 전송 | 암호화되지 않은 평문 데이터를 탈취하여 중요한 정보를 획득할 수 있는 보안취약점 | 중요 정보를 저장하거나 전송할 때는 반드시 암호화 과정을 거치도록 하고, HTTPS 또는 SSL/TLS 등의 보안 채널을 이용하여 방지 |
| 하드코딩된 비밀번호 | 프로그램 코드 내부에 패스워드 포함 시 관리자 정보가 노출될 수 있는 보안 취약점 | <ul style="list-style-type: none"> - 패스워드는 암호화하여 별도 파일에 저장 - 소프트웨어 설치 시 직접 패스워드나 키를 입력하도록 설계하여 방지 |
| 취약한 패스워드 요구조건 | 취약한 사용자 패스워드 조합 규칙에 따른 사용자 계정 보안 취약점 | 패스워드 생성 시 강한 조건 검증 필요 |

• 에러 처리

| 보안 취약점 | 설명 | 대책 |
|-----------------|---|---|
| 오류 메시지 통한 정보 노출 | 프로그램이 실행환경, 사용자 정보, 디버깅 정보 등의 중요 정보를 포함하는 오류 메시지를 생성하여 공격자의 악성 행위를 도와주는 보안 취약점 | 오류 발생 시 가능한 한 내부에서만 처리되도록 하거나 메시지를 출력할 경우 최소한의 정보 또는 사전에 준비된 메시지만 출력되도록 함으로써 방지 |
| 오류 상황 대응 부재 | 오류가 발생할 수 있는 부분에 대해 예외처리를 하지 않았거나 예외처리 미비로 인해 발생할 수 있는 보안 취약점 | 오류가 발생할 수 있는 부분에 예외처리 구문을 작성하고, 제어문을 활용하여 오류가 악용되지 않도록 코딩함으로써 방지 |
| 부적절한 예외 처리 | 프로그램 수행 중에 함수의 반환값 또는 오류들을 세분화하여 처리하지 않고 광범위하게 묶어 한 번에 처리하거나, 누락된 예외가 존재할 때 발생하는 보안 취약점 | 모든 함수의 반환값이 의도대로 출력되는지 확인하고, 광범위한 예외처리 대신 구체적인 예외처리를 통해 방지 |

• 세션 통제(Session Control)

| 보안 취약점 | 설명 | 대책 |
|------------|--|---|
| 불충분한 세션 관리 | <ul style="list-style-type: none"> - 인증 시 일정한 규칙이 존재하는 세션ID가 발급되거나, 세션 타임아웃이 너무 길게 설정되어 있는 경우 발생할 수 있는 보안 취약점 - 세션 관리가 충분하지 않으면 침입자는 세션 하이재킹과 같은 공격을 통해 획득한 세션ID로 인가되지 않은 시스템의 기능을 이용하거나 중요한 정보에 접근 가능 | <ul style="list-style-type: none"> - 세션 ID의 예측이 불가능하도록 안전한 난수 알고리즘 적용 - 로그인 시 로그인 전의 세션 ID를 삭제하고 재할당 - 장기간 접속하고 있는 세션ID는 주기적으로 재할당하도록 설계 |

• 코드 오류

| 보안 취약점 | 설명 | 대책 |
|--------------------------|--|---|
| 널 포인터 (Null Pointer) 역참조 | <ul style="list-style-type: none"> - 널 포인터가 가리키는 메모리에 어떠한 값을 저장할 때 발생하는 보안 취약점 - 많은 라이브러리 함수들이 오류가 발생할 경우 Null 값을 반환하는데, 이 반환값을 포인터로 참조하는 경우 발생 - 대부분 운영체제에서 널 포인터는 메모리의 첫 주소를 가리키며, 해당 주소를 참조할 경우 소프트웨어가 비정상적으로 종료될 수 있음 - 공격자가 의도적으로 널 포인터 역참조를 실행하는 경우, 그 결과 발생하는 예외사항을 추후에 공격자가 악용할 수 있음 | <ul style="list-style-type: none"> - Null이 될 수 있는 포인터를 이용하기 전에 Null 값을 갖고 있는지 검사한 후 안전한 경우에만 사용 - 스택 가드(Stack Guard) 활용 |
| 부적절한 자원 해제 | <ul style="list-style-type: none"> - 부적절한 자원 해제는 자원을 반환하는 코드를 누락하거나 프로그램 오류로 할당된 자원을 반환하지 못했을 때 발생하는 보안 취약점 - 힙 메모리(Heap Memory), 소켓(Socket) 등의 유한한 시스템 자원이 계속 점유하고 있으면 자원 부족으로 인해 새로운 입력을 처리하지 못할 수 있음 | 자원을 획득하여 사용한 다음에는 Finally 블록에서 반드시 자원이 반환되도록 코딩함으로써 방지 |
| 해제된 자원 사용 | <ul style="list-style-type: none"> - 이미 사용이 종료되어 반환된 메모리를 참조하는 경우 발생하는 보안 취약점 - 반환된 메모리를 참조하는 경우 예상하지 못한 값 또는 코드를 수행하게 되어 의도하지 않은 결과가 발생할 수 있음 | 반환된 메모리에 접근할 수 없도록 주소를 저장하고 있는 포인터를 초기화함으로써 방지 |
| 초기화되지 않은 변수 사용 | <ul style="list-style-type: none"> - 변수 선언 후 값이 부여되지 않은 변수를 사용할 때 발생할 수 있는 보안 취약점 - 변수가 선언되어 메모리가 할당되면 해당 메모리에 이전에 사용하던 내용이 계속 남아 있어 변수가 외부에 노출되는 경우 중요정보가 악용될 수 있음 | 변수 선언 시 할당된 메모리를 초기화함으로써 방지 |
| 정수를 문자로 변환 | 정수를 문자로 변환하면서 표현할 수 없는 범위의 값이 잘려나가 문자에 대한 저장 값이 올바르지 않은 보안 취약점 | 정수를 문자로 변환할 경우, 변환 값의 크기가 변환 값이 저장되는 변수의 크기보다 크지 않도록 함 |

• 캡슐화

| 보안 취약점 | 설명 | 대책 |
|------------------------------|--|---|
| 잘못된 세션에 의한 데이터 정보 노출 | <ul style="list-style-type: none"> - 다중 스레드(Multi-Thread) 환경에서 멤버 변수에 정보를 저장할 때 발생하는 보안 취약점 - 싱글톤(Singleton) 패턴에서 발생하는 경쟁 조건(Race Condition)으로 인해 동기화 오류가 발생하거나, 멤버 변수의 정보가 노출될 수 있음 | 싱글톤 패턴을 사용할 경우 변수 범위(Scope)에 주의하고, 멤버 변수보다 지역 변수를 활용해 변수의 범위를 제한함으로써 방지 |
| 제거되지 않고 남은 디버그 코드 | <ul style="list-style-type: none"> - 개발 중에 버그 수정이나 결과값 확인을 위해 남겨둔 코드들로 인해 발생하는 보안 취약점 - 소프트웨어 제어에 사용되는 중요한 정보가 디버그 코드로 인해 노출될 수 있음 - 디버그 코드에 인증 및 식별 절차를 생략하거나 우회하는 코드가 포함되어 있는 경우 공격자가 이를 악용할 수 있음 | 디버그 코드는 개발 완료 후 삭제 처리 |
| Public 메소드로부터 반환된 Private 배열 | <ul style="list-style-type: none"> - 선언된 클래스 내에서만 접근이 가능한 Private 배열을 모든 클래스에서 접근이 가능한 Public 메소드에서 반환할 때 발생하는 보안 취약점 - Public 메소드가 Private 배열을 반환하면 배열의 주소가 외부로 공개되어 외부에서 접근할 수 있게 됨 | Private 배열을 별도의 메소드를 통해 조작하거나, 동일한 형태의 복제본으로 반환받은 후 값을 전달하는 방식으로 방지 |
| 민감한 데이터를 가진 내부 클래스 사용 | 권한이 없는 클래스를 사용하고자 할 때 발생하는 보안 취약점 | 내부 클래스 사용 시 외부 클래스의 접근 금지 |
| 시스템 데이터 정보 노출 | 시스템의 내부 정보를 시스템 메시지 등을 출력하도록 코딩했을 때 발생하는 보안 취약점 | 시스템 메시지를 통해 노출되는 메시지는 최소한의 정보만을 제공함으로써 방지 |

• API 오용

| 보안 취약점 | 설명 | 대책 |
|-----------------------|---|--|
| DNS lookup에 의존한 보안 결정 | <ul style="list-style-type: none"> - 도메인명에 의존하여 인증이나 접근 통제 등의 보안 결정을 내리는 경우 발생하는 보안 취약점 - DNS 엔트리를 속여 동일한 도메인에 속한 서버인 것처럼 위장하거나, 사용자와 서버 간의 네트워크 트래픽을 유도하여 악성 사이트를 경유하도록 조작할 수 있음 - 공격자는 DNS lookup을 악용하여 인증이나 접근 통제를 우회하는 수법으로 권한을 탈취함 | DNS 검색을 통해 도메인 이름을 비교하지 않고 IP 주소를 직접 입력하여 접근함으로써 방지 |
| 취약한 API 사용 | <ul style="list-style-type: none"> - 보안 문제로 사용이 금지된 API를 사용하거나, 잘못된 방식으로 API를 사용했을 때 발생하는 보안 취약점 - 보안 문제로 금지된 대표적인 API에는 C언어의 문자열 함수 strcpy(), strncpy(), sprintf() 등이 있음 - 보안 상 안전한 API라고 하더라도 자원에 대한 직접 연결이나, 네트워크 소켓을 통한 직접 호출과 같이 보안에 위협을 줄 수 있는 인터페이스를 사용하는 경우 보안 약점이 노출됨 | <p>보안 문제로 금지된 함수는 안전한 함수로 대체하고, API의 매뉴얼을 참고하여 보안이 보장되는 인터페이스를 사용함으로써 방지 (예) strcpy(): 글자 수 상관없이 문자열 복사 가능하여 위험 → strncpy(): 문자열 복사 시 글자 길이 지정</p> |

106 암호 알고리즘★★★

• 암호 알고리즘 관련 용어

| 용어 | 설명 |
|-------------------------------------|---|
| 평문 (Plain) | 암호화되기 전의 원본 메시지 |
| 암호문(Cipher) | 암호화가 적용된 메시지 |
| 암호화(Encryption) | 평문을 암호문으로 바꾸는 작업 |
| 복호화(Decryption) | 암호문을 평문으로 바꾸는 작업 |
| 키(Key) | 적절한 암호화를 위하여 사용하는 값 |
| 치환 암호(대치암호: Substitution Cipher) | 비트, 문자 또는 문자의 블록을 다른 비트, 문자 또는 블록으로 대체하는 방법 |
| 전치 암호(Transposition Cipher) | 비트, 문자 또는 블록이 원래 의미를 감추도록 자리바꿈 등을 이용하여 재배열하는 방법 |

• 양방향 암호화 알고리즘 - 대칭키/비대칭 키 암호 방식 비교

| 구분 | 대칭 키 암호 방식 | 비대칭 키 암호 방식 |
|-------|--|--|
| 키 | 대칭 키(=개인 키, 비밀 키) | 비대칭 키(=공개 키) |
| 키의 관계 | 암호화 키=복호화 키 | 암호화 키≠복호화 키 |
| 암호화 키 | 비밀 키 | 공개 키 |
| 복호화 키 | 비밀 키 | 개인 키 |
| 키 개수 | $\frac{n(n-1)}{2}$ 10명이 공개키 암호를 사용할 경우 45개의 키가 필요하다. | $2n$ 10명이 공개키 암호를 사용할 경우 20개의 키가 필요하다. |
| 장점 | - 암호복호화 키 길이가 짧음 - 암호복호화 속도가 빠름 | - 암호화 키 사전 공유 불필요 - 관리해야 할 키 개수가 적음 - 키 분배 및 관리가 쉬움 - 개인 키 활용해 인증, 전자 서명 등에 적용 가능 |
| 단점 | - 키 분배 및 관리의 어려움 - 기밀성만 보장 | - 암호복호화 키 길이가 김 - 암호복호화 속도가 느림 |
| 알고리즘 | - 블록 암호화 방식: DES, SEED, AES, ARIA, IDEA - 스트림 암호화 방식: LFSR, RC4 | 디피-헬만(Diffie-Hellman), RSA, ECC, Elgamal, DAS |

107 서비스 공격 기법★★

• 서비스 공격 유형과 공격도구/탐지 기법

| 구분 | 공격기법 | 설명 |
|-------------|-------------------------------|--|
| 정보 보안 침해 공격 | Buffer Overflow (버퍼 오버플로우) | 메모리에 할당된 버퍼 크기를 초과하는 양의 데이터를 입력해 프로세스의 흐름을 변경시켜 악성코드를 실행시키는 공격 기법 |
| | Backdoor (백도어) | - ‘뒷문(Backdoor)’이라는 단어의 어감에서 알 수 있듯, 어떤 제품이나 컴퓨터 시스템, 암호시스템 또는 알고리즘에서 정상적인 인증 절차를 우회하는 기법. 즉, 허가받지 않고 시스템에 접속하는 권리를 얻음 - 해커는 백도어를 통해서 이용자 몰래 컴퓨터에 접속하여 악의적인 행위를 하기도 함 - 백도어 탐지기법: 프로세스 및 열린 포트 확인, SetUid 파일 검사, 백신 및 백도어 탐지 툴 활용, 무결성 검사, 로그 분석 |

| | |
|--|--|
| Key Logger Attack (키로거 공격) | 컴퓨터 사용자의 키보드 움직임을 탐지해서 저장하고, ID나 패스워드, 계좌 번호, 카드 번호 등과 같은 개인의 중요한 정보를 몰래 빼 가는 공격 기법 |
| Format String Attack (포맷 스트링 공격) | 포맷 스트링을 인자로 하는 함수의 취약점을 이용한 공격으로 외부로부터 입력된 값을 검증하지 않고 입출력 함수의 포맷 스트링을 그대로 사용하는 경우 발생 |
| Race Condition Attack (레이스 컨디션 공격) | 레이스 컨디션 공격은 실행되는 프로세스가 임시파일을 만드는 경우 악의적인 프로그램을 통해 그 프로세스의 실행 중에 끼어들어 임시파일을 심볼릭 링크하여 악의적인 행위를 수행하게 하는 공격 기법 |
| Rootkit (루트킷) | 시스템 침입 후 침입 사실을 숨긴 채 차후의 침입을 위한 백도어, 트로이 목마 설치, 원격 접근, 내부 사용 흔적 삭제, 관리자 권한 획득 등 주로 불법적인 해킹에 사용되는 기능을 제공하는 프로그램의 모음 |
| Phishing (피싱) | 소셜 네트워크에서 악의적인 사용자가 지인 또는 특정 유명인으로 가장해 불특정 다수의 정보를 탈취하는 공격 기법 |
| Spear Phishing (스피어피싱) | (공격) 사회 공학의 한 기법으로, 특정 대상을 선정한 후 그 대상에게 일반적인 이메일로 위장한 메일을 지속적으로 발송하여, 발송 메일의 본문 링크나 첨부된 파일을 클릭하도록 유도하여 사용자의 개인정보를 탈취하는 공격 기법(사이버 사기) |
| Smishing (스미싱) | 문자메시지를 이용해 신뢰할 수 있는 사람 또는 기업이 보낸 것처럼 가장해 개인 비밀정보를 요구하거나 휴대폰 소액 결제를 유도하는 피싱 공격(사이버 사기) |
| Qshing (큐싱) | 스마트폰을 이용하여 금융 업무를 처리하는 사용자에게 인증 등이 필요한 것처럼 속여 QR 코드를 통해 악성 앱을 내려받도록 유도, 금융 정보 등을 빼내는 피싱 공격(사이버 사기) |
| Evil Twin Attack (이블 트윈 공격) | 무선 Wifi 피싱 기법으로 공격자는 합법적인 Wifi 제공자처럼 행세하며 노트북이나 휴대 전화로 핫스팟에 연결한 무선 사용자들의 정보를 취하는 무선 네트워크 공격 기법 |
| Worm (웜) | 다른 컴퓨터의 취약점을 이용해 스스로 전파하거나 메일로 전파되며 스스로를 증식하는 악성 소프트웨어 컴퓨터 프로그램 혹은 코드 |
| Malicious Bot (악성 봇) | - 스스로 실행되지 못하고, 해커의 명령에 의해 원격에서 제어 또는 실행이 가능한 프로그램 혹은 코드 - 주로 취약점이나 백도어 등을 이용하여 전파되며, 스팸 메일 전송이나 분산 서비스 거부 공격(DDoS) 등에 악용 |
| Zombie(좀비) PC | - 악성 봇에 의해 감염된 PC - C&C(Command & Control) 서버의 제어를 받아 주로 DDoS 공격 등에 이용 |
| C&C 서버 | 해커가 원격지에서 감염된 좀비 PC에 명령을 내리고 악성코드를 제어하기 위한 용도로 사용하는 서버 |
| Botnet (봇넷) | 악성 프로그램에 감염되어 악의적인 의도로 사용될 수 있는 다수의 컴퓨터들이 네트워크로 연결된 형태 |
| Ransomware (랜섬웨어) | 인터넷 사용자의 컴퓨터에 침입해 내부 문서 파일 등을 암호화해 사용자가 열지 못하게 하는 공격 기법 |
| Logic Bomb (논리 폭탄) | 특정 날짜나 시간 등 조건이 충족되었을 때 악의적인 기능(Function)을 유발할 수 있게 만든 코드의 일부분으로 소프트웨어 시스템에 의도적으로 삽입된 악성 코드 |
| Advanced Persistent Threat (APT 공격) | 다양한 IT 기술과 방식들을 이용해 조직적으로 특정 기업이나 조직 네트워크에 침투해 활동 거점을 마련한 뒤 때를 기다리면서 보안을 무력화시키고 정보를 수집한 다음 외부로 빼돌리는 형태의 공격 |
| Supply Chain Attack (공급망 공격) | 소프트웨어 개발사의 네트워크에 침투하여 소스 코드에 악의적인 코드를 삽입하거나 배포 서버에 접근해 악의적인 파일로 변경하는 방식을 통해 사용자 PC에 소프트웨어를 설치 또는 업데이트 시에 자동적으로 감염되도록 하는 공격 |

| | | |
|---------------|--|---|
| | Zero Day Attack (제로데이 공격) | 보안 취약점이 발견되어 널리 공표되기 전에 해당 취약점을 악용하여 이루어지는 보안 공격 |
| | Trojan Horse (트로이 목마) | 악성 루틴이 숨어 있는 프로그램으로 겉보기에는 정상적인 프로그램으로 보이지만 실행하면 악성 코드를 실행 |
| 네트워크 침해 공격 | Switch Jamming (스위치 재밍) | 위조된 매체 접근 제어(MAC) 주소를 지속적으로 네트워크로 흘려보내, 스위치 MAC 주소 테이블의 저장 기능을 혼란시켜 더미 허브(Dummy Hub)처럼 작동하게 하는 공격 |
| | Sniffing (스니핑) | 암호화되지 않은 패킷들을 수집하여 순서대로 재조합 후 ID, PW와 같은 중요한 정보를 유출하기 위한 수동적인 형태의 공격 |
| | Network Scanner / Sniffer (네트워크 스캐너 / 스니퍼) | (도구) 네트워크 하드웨어 및 소프트웨어 구성의 취약점 파악을 위해 공격자가 사용하는 공격 도구 |
| | Password Cracking (패스워드 크래킹) | 패스워드를 '깨뜨리다(Crack)'는 말로, 공격자가 암호화된 사용자의 패스워드를 평문 형태로 알아내는 공격 (Dictionary Attack, Brute Force Attack, Password Hybrid Attack, Rainbow Table Attack) |
| | IP Spoofing (IP 스푸핑) | <ul style="list-style-type: none"> - 서버에 대한 인증되지 않은 액세스 권한을 입수하는 데 사용하는 기법 - 침입자가 패킷 헤더 수정을 통해 인증된 호스트의 IP 주소를 위조 - 타깃 서버로 메시지를 발송한 이후 패킷은 해당 포트에서 유입되는 것처럼 표시 |
| | ARP Spoofing (ARP 스푸핑) | 공격자가 특정 호스트에게 잘못된 MAC 주소가 담긴 ARP Reply를 보내, 호스트의 ARP 캐시를 조작하여 호스트로부터 정보를 빼내는 공격 기법 |
| | ICMP Redirect 공격 | ICMP Redirect 메시지를 공격자가 원하는 형태로 만들어서 특정 목적지로 가는 패킷을 공격자가 스니핑하는 기법 |
| | Session Hijacking (세션 하이재킹) | <ul style="list-style-type: none"> - 서버에 접속하고 있는 클라이언트들의 세션 정보를 가로채는 공격 기법 - 세션 하이재킹의 탐지 방법: 비동기화 상태 탐지, ACK 패킷 비율 모니터링, 패킷의 유실 탐지, 예상치 못한 접속의 리셋 탐지 |
| 블루투스 관련 공격 | BlueBug (블루버그) | 블루투스 장비 사이의 취약한 연결 관리를 악용한 공격으로 휴대폰 원격 조정 또는 통화 감청 |
| | BlueSnarf (블루스나프) | 블루투스의 취약점을 활용하여 장비의 파일에 접근하는 공격으로 인증 없이 간편하게 정보를 교환할 수 있는 OPP(Object Push Profile)를 사용하여 정보 열람 |
| | Blueprinting (블루프린팅) | 공격 대상이 될 블루투스 장비를 검색하는 활동 |
| | BlueJacking (블루재킹) | 블루투스를 이용해 스팸처럼 메시지를 익명으로 퍼뜨리는 공격 |

108 서버 인증★

• 인증 기술의 유형

| 유형 | 설명 | 예시 |
|--------------|---|-----------------------|
| 지식 기반 인증 | 주체가 '알고 있는 것'(Something you know)을 보여 주며 인증 | 패스워드, PIN 번호 등 |
| 소유 기반 인증 | 주체가 '그가 가지고 있는 것'(Something you have)을 보여 주며 인증 | 토큰, 스마트카드, 신분증, OTP 등 |
| 생체(존재) 기반 인증 | 주체가 '그가 가지고 있는 고유한 생체적 특징'(Something You Are)을 보여 주며 인증 | 홍채, 지문, 얼굴 등 |
| 행위 기반 인증 | 주체가 '그가 하는 것'(Something you do)을 보여 주며 인증 | 서명, 발걸음, 몸짓 등 |

109 정보보안과 접근 제어 ★★★

• 접근 제어 정책

| 구분 | DAC (Discretionary Access Control) | MAC (Mandatory Access Control) | RBAC (Role Based Access Control) |
|--------|---------------------------------------|-----------------------------------|-------------------------------------|
| 의미 | 신분 기반(임의적) | 규칙 기반(강제적) | 역할 기반 |
| 권한 부여자 | 접근제어 정책 | 접근제어 정책 | 접근제어 정책 |
| 접근 결정 | 데이터 소유자 | 시스템 | 중앙관리자 |
| 정책 변경 | 신분(Identity) | 보안등급(Label) | 역할(Role) |
| 장점 | 변경 용이 | 고정적(변경 어려움) | 변경 용이 |
| | 구현 용이, 유연함 | 안정적, 중앙 집중적 | 관리 용이 |

110 시스템 보안 구현 ★★

• 리눅스(LINUX)의 커널 로그

| 데몬 | 파일명 | 내용 |
|--------|-----------------|---|
| kernel | /dev/console | 커널에 관련된 내용을 관리자에게 알리기 위해 파일로 저장하지 않고 지정된 장치에 표시 |
| | var/log/wtmp | - 성공한 로그인/로그아웃에 대한 로그를 기록 - 시스템의 시작/종료 시간에 대한 로그를 기록 |
| | var/run/utmp | 현재 로그인한 사용자의 상태에 대한 로그를 기록 |
| | var/log/btmp | 실패한 로그인에 대한 로그를 기록 |
| | var/log/lastlog | 마지막으로 성공한 로그인에 대한 로그를 기록 |

• 네트워크 보안 솔루션

| 솔루션 | 설명 |
|--|---|
| 방화벽 (Firewall) | 기업 내부, 외부 간 트래픽을 모니터링하여 시스템의 접근을 허용하거나 차단하는 시스템 |
| 웹 방화벽 (WAF: Web Application Firewall) | - 일반적인 네트워크 방화벽과는 달리 웹 애플리케이션 보안에 특화된 보안장비 - SQL 인젝션, XSS 등과 같은 웹 공격을 탐지하고 차단 |
| 네트워크 접근 제어 (NAC: Network Access Control) | - 단말기가 내부 네트워크에 접속을 시도할 때 이를 제어하고 통제하는 기능을 제공 - 바이러스나 웜 등의 보안 위협으로부터 네트워크 제어 및 통제 기능을 수행 |
| 침입 탐지 시스템 (IDS: Intrusion Detection System) | 네트워크에서 발생하는 이벤트를 모니터링하고 비인가 사용자에 의한 자원 접근과 보안정책 위반 행위(침입)를 실시간으로 탐지하는 시스템 |

| | |
|--|---|
| 침입 방지 시스템 (IPS: Intrusion Prevention System) | 네트워크에 대한 공격이나 침입을 실시간적으로 차단하고, 유해 트래픽에 대한 조치를 능동적으로 처리하는 시스템 |
| 무선 침입 방지 시스템 (WIPS: Wireless Intrusion Prevention System) | <ul style="list-style-type: none"> - 인가되지 않은 무선 단말기의 접속을 자동으로 탐지 및 차단하고 보안에 취약한 무선 공유기를 탐지하는 시스템 - 외부 공격에 대해 내부 시스템을 보호하기 위해 무선 랜 환경에서의 보안 위협을 감지 |
| 통합 보안 시스템 (UTM: Unified Threat Management) | 방화벽, 침입 탐지 시스템(IDS) 침입 방지 시스템(IPS), VPN, 안티 바이러스, 이메일 필터링 등 다양한 보안 기능을 하나의 장비로 통합하여 제공하는 시스템 |
| 가상사설망 (VPN: Virtual Private Network) | 인터넷과 같은 공중망에 사설망을 구축하여 마치 전용망을 사용하는 효과를 가지는 보안 솔루션 |

