Software Security

Security Development Process for KUPE

IT융합 정보보호학과

조 원 : 두 상 균
안 정 현
발 표 자 : 이 남 곤

Security

윈도우 취약점 공개한 구글에 MS는 "실망스럽다"

어도비, 플래시 플레이어 위한 긴급 보안 패치 발표

'오래된' 오픈SSH 취약점과 '최신' IoT 결합...신종 사이버공격 공포

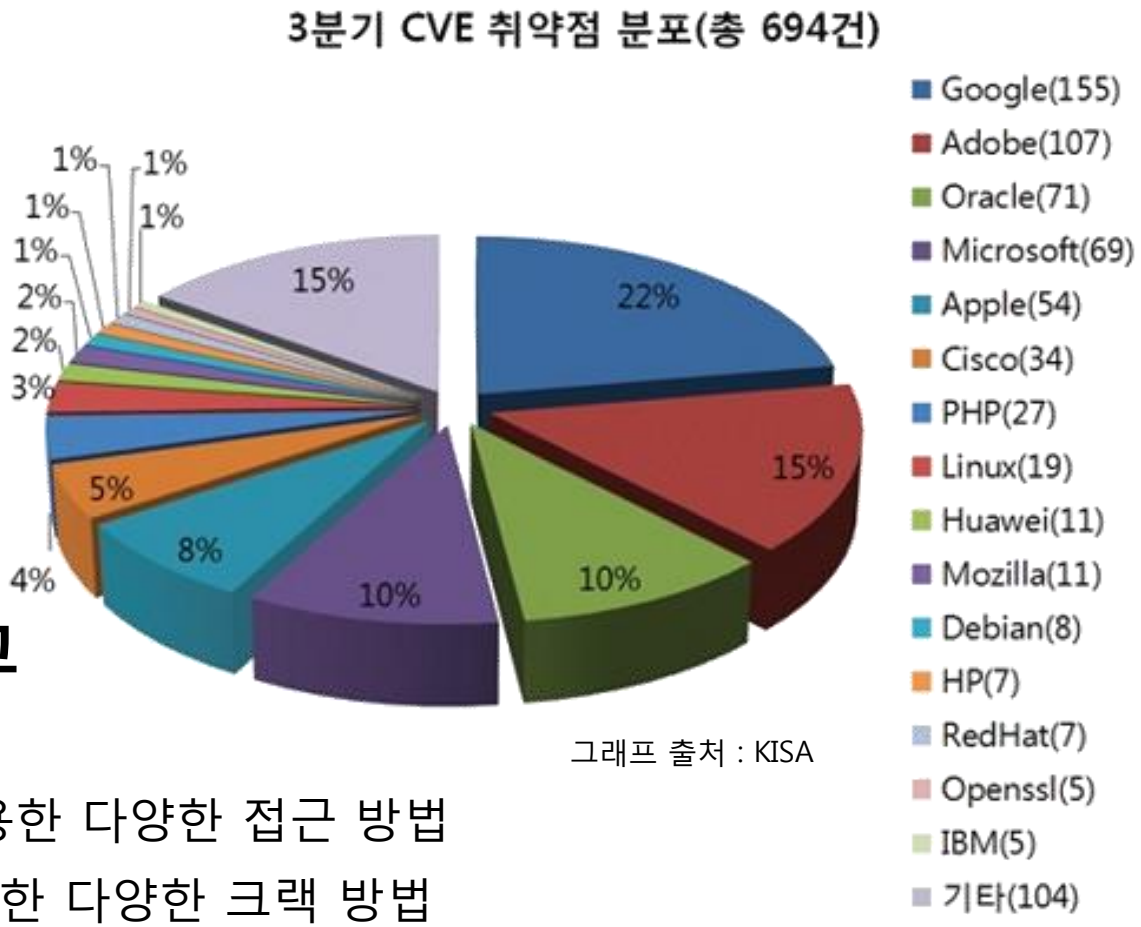
닭이 먼저냐, 달걀이 먼저냐, 취약점과 보안 업데이트

패치도 안 되는 윈도우 OS 공격방법 개발되다

-
-
-
-

기사 출처 : 보안 뉴스

Security



그래프 출처 : KISA

끊이지 않는 보안 사고

- 고도화된 통신 기술을 이용한 다양한 접근 방법
- 강력해진 퍼포먼스를 기반한 다양한 크랙 방법
- 소프트웨어의 다양성으로 인한 다양한 취약점 산재
- 사람들의 안이한 보안 인식

Select

보안 취약점 중 중요도 선택 중요

- 단순 장난, 서비스 방해는 비즈니스에 크리티컬 하지 않음
- 자사 정보의 유출 및 삭제, 변경은 크리티컬
- 고도로 중요한 정보일 수록 접근이 제한적
- 루트 권한 탈취 시 모든 정보의 접근 및 삭제 가능

Vulnerability

SQL Injection

- 입력란에 쿼리문을 삽입하여 데이터베이스에 접근하는 방법
- DB Admin 권한 탈취 및 DB 변경 가능

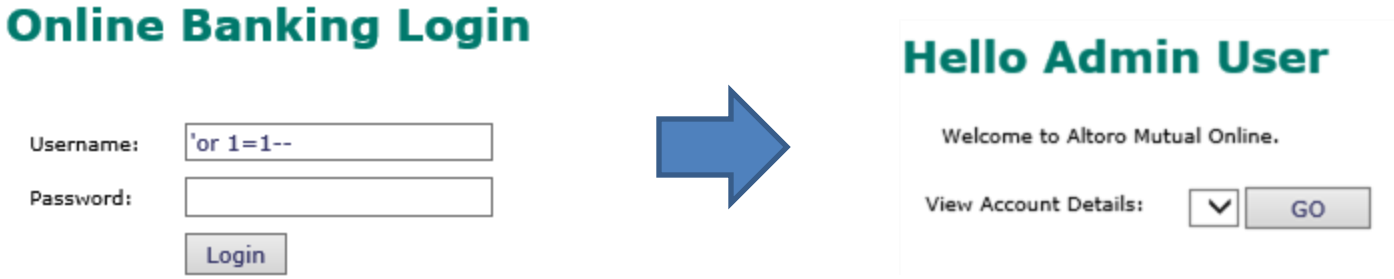
Buffer Overflow

- 회귀 주소 조작으로 특정 명령어 및 메모리에 접근하게 하는 방법
- 권한 탈취 및 권한이 없는 프로그램 실행 가능

SQL Injection

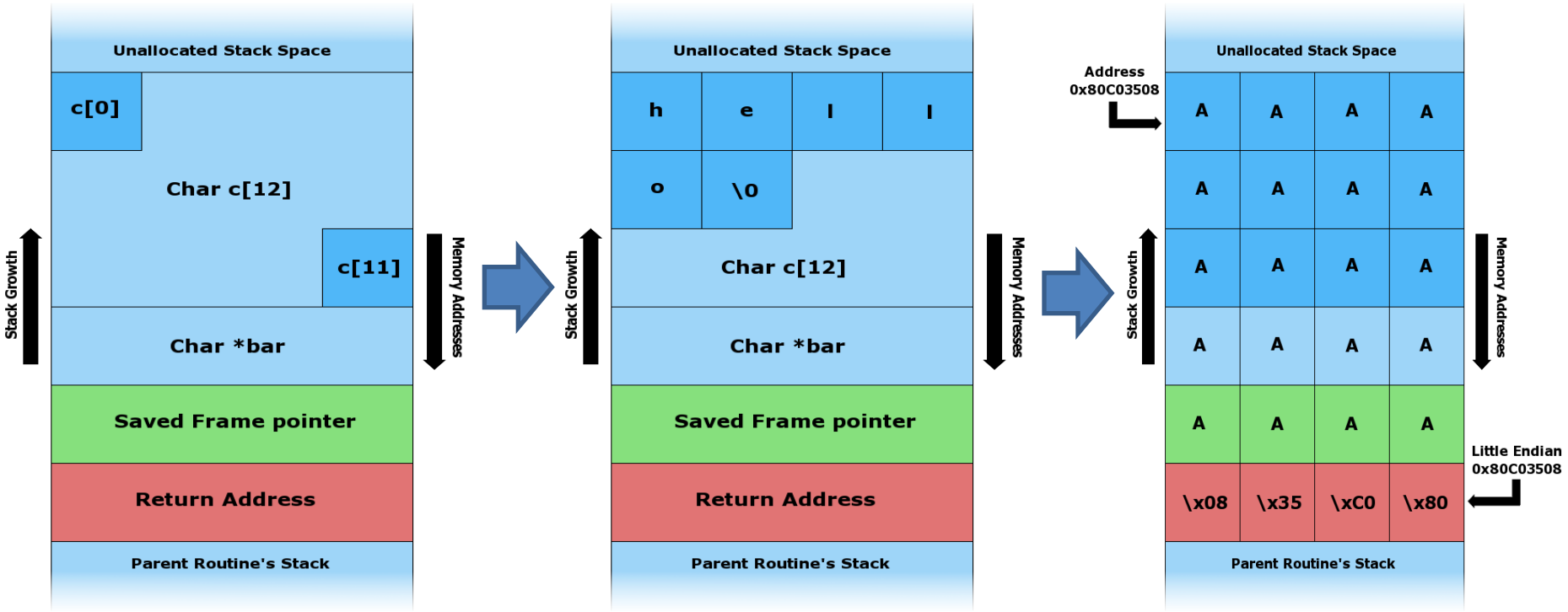
select user_id, user_pw form member where user_id='"id"' and user_pw='"password"'

or 1=1	or문을 이용한 공격, 1=1이 참이면 True
or 2>1	2가 1보다 크다가 참이면 True
or 'abc'='abc'	'abc'와 'abc'가 같으면 true
--	-- 뒤의 구문 주석 처리



우회 패턴과 주석 구문을 이용한 접속 예제

Buffer Overflow



Resolve

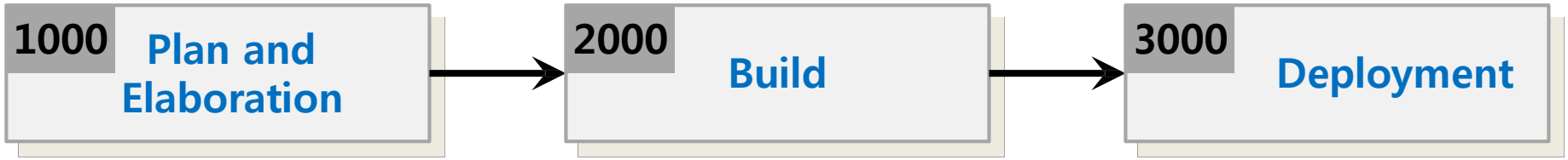
SQL Injection

- DB 회사에서 제공하는 Prepared Statement 사용
- DB 관련 페이지의 에러 페이지 노출 되지 않도록 작성
- 웹 방화벽 사용

Buffer Overflow

- 버퍼 크기를 체크하여 해당 크기를 넘는 데이터 삽입 방지
- Return Address 값의 변조 발견 시 강제 종료
- 문자열 사용시 Null 값으로 종료 강제

Process



각 파트별 부분 수정

- 보안 요소는 비즈니스 모델에 큰 영향을 주지 않음
- 사용자에게 드러내 보일 필요 없음
- 필요한 부분이기에 퍼포먼스 감소에 대해 어느 정도의 관대함이 작용
- 보안 사고 사례 및 CVE를 통한 구체적 예 확보 가능

Phase 1002. Create Preliminary Investigation Report

- Description
  - Write an investigation report on alternatives, business needs, risk, etc
  - Input : draft project plan
  - Output : an investigation report
- Steps
  1. Write alternative solutions
  2. Write project's justification (business needs)
  3. Identify and manage risks, and write risk reduction plans
  4. Analyze business market
  5. Write managerial issues

Phase 1002. Create Preliminary Investigation Report

Risk Management

Risk	Probability	Significance	Weight
Buffer Overflow	-	5	20
SQL Injection	-	5	20

Risk Reduction Plan

- Buffer Overflow (20)
  입력 받는 값을 정확히 하고 해당 버퍼의 크기를 정확히 지정

- SQL Injection (20)
  SQL의 입력 값 대한 필터링 및 에러 메시지가 외부로 유출 되지 않도록 주의

Phase 1003. Define Requirements

- Description
  - Write a requirement specification for a product
  - Input : draft project plan, investigation report
  - Output : a requirement specification
- What is a requirement? (IEEE Std 610.12-1990)
  - A condition or capability needed by a user to solve a problem or achieve an objective.
  - A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents.
  - A documented representation of a condition or capabilities as in (1) or (2)

Phase 1003. Define Requirements

Security Functions

Ref. #	Vulnerability	Functions	Category
R1	Buffer Overflow	Buffer size check Return address check Etc	Secure Coding
R2	SQL Injection	SQL filtering Error page block Etc	Secure Coding

Phase 1008. Define Draft System Architecture

- Description
  - Construct a rough preliminary system architecture model
  - Input : requirements specification business use case model
  - Output : a draft system architecture
- Steps
  1. Define logical/physical layers of the target system
  2. Separate the whole system into several subsystems
  3. Assign business use cases into each subsystem
  4. Identify and draw up hardware resources

Phase 1008. Define Draft System Architecture

Security Architecture

Activity 2030(sec). Check Possible Use Case Vulnerability

- 해당 시스템에서 발생할 수 있는 보안 취약점을 고려
- 취약점 유형 분류는 다음과 같이 나뉨
  a. 입력데이터 검증 및 표현
     - 입력값에 대한 검증 누락, 잘못된 데이터 형식지정 등
     - ex) Buffer Overflow, SQL Injection, XSS 등
  b. 보안기능(인증, 접근제어, 기밀성, 암호화, 권한관리 등)의 부적절 구현
     - ex) 부적절한 인가, 중요정보 평문저장, 하드코드된 패스워드
  c. 에러처리
     - 불충분한 처리로 에러정보에 중요정보가 포함될 때 발생 가능한 약점
     - ex) 오류 메시지를 통한 정보노출
  d. 캡슐화
     - 중요한 데이터의 캡슐화 보안성이 낮아 비인가자에게 데이터 누출이 가능해지는 약점
     - ex) 제거되지 않고 남은 디버거코드, 시스템데이터 정보노출 등

참조) 붙임1_소프트웨어_개발보안_가이드.pdf

Activity 2030(sec). Check Possible Use Case Vulnerability

- 그 중 권한탈취의 주요 원인이 되는 입력데이터 검증 및 표현 취약점들에 대해 예시(Buffer Overflow, SQL injection)
- 입력 : Business use case description(1006)
- 출력 : Security requirement description

- Step
  a. Select each use case from business use cases
  b. Identify system functions corresponded to each use case
  c. Find system functions that user input data

Ref. #	Function	Use Case Number & Names	Category
R 1.1	System Access	1. Login	Primary
R 1.1	System Access	2. Logout	Primary
R 1.2	Make Account	3. Make Account	Primary
R 1.3	Identify Balance	4. Identify Balance	Primary
R 1.4	Recharge Balance	5. Recharge Balance	Primary
R 2.1	Request Print	6. Request Print	Primary
R 2.2	Check Balance	7. Check Balance	Primary
R 3.1	Identify Paper	8 Identify Paper	Primary
R 3.2	Recharge Paper	9. Recharge Paper	Primary
R 3.3	Identify User	10. Identify User	Primary
R 3.4	Identify Money	11. Identify Money	Primary

  d. Check whether upper vulnerabilities could be occurred
     - Buffer Overflow / SQL injection
  e. Write security requirement description based on possible vulnerabilities of step d

Activity 2030(sec). Check Possible Use Case Vulnerability

➢ Security requirement description format

Use Case	Login
Actors	User, Manager
Description	- 사용자 or 관리자가 시스템을 사용하기 위해 id, pw 를 입력 받는다. - 이 use case는 입력 받은 id/pw 와 저장된 회원 정보를 확인하여 id, pw가 일치하는 경우 시스템 사용을 승인 한다. - 회원인 경우 user로 관리자인 경우 manager로 로그인 하고 화면을 전환한다.

- id, pw가 일치하는 경우 그 데이터 형식을 확인
(SQL query문 형태인지, PL/SQL 함수 형태인지, 입력값이 비정상적으로 긴지)

Use Case	Make Account
Actors	User, Manager
Description	- 이 use case는 id/pw 를 입력 받는다 - 입력 받은 id 와 일치하는 id가 없는 경우 사용자 계정을 생성한다.

- 입력 id가 기존 db에 없는 경우 그 데이터 형식을 확인
(SQL query문 형태인지, PL/SQL 함수 형태인지, 입력값이 비정상적으로 긴지)

Phase 2030. Analyze

2031. Define Essential Use case(필수 유즈케이스 다이어그램 정의)

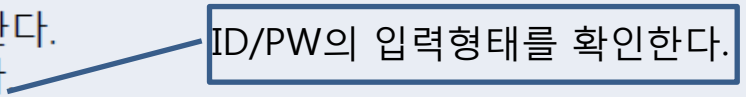
- Business use case description에 event flow를 추가함
- 입력 : Business use case description(1006), ***Security requirement description***
- 출력 : An essential use case description ***(Secure use case description)***
- 적용표준 : 확장된 use case format

- Step
  a. Business use case에서 각각의 use case 선택
  b. 요구사항 명세서에서 선택된 use case와 관련이 있는 system함수 식별
  c. Business use case에서 step b를 통해 선택된 use case 식별
  d. 요구사항 명세서와 관련이 있는 각각 use case의 이벤트 과정을 식별
     (이 부분에 보안 이벤트 추가)
     - ✓ 주요 이벤트 과정(main event flow)
     - ✓ 대체 이벤트 과정(우회 처리)
     - ✓ 예외 이벤트 과정(예외 처리)
  a. 주요 및 대체 이벤트 과정을 토대로 essential use case 작성
     (확장된 use case format을 적용함)

Phase 2030. Analyze

➢ Example : Buy Items(위 항목을 보고 해당 표의 내용과 대조해보세요)

Use Case	Make Account
Actor	User
Purpose	새로운 사용자 계정을 생성한다.
Overview	생성할 계정의 id/pw를 입력 후 생성 버튼을 눌러 새로운 사용자 계정을 생성 할 수 있다. 새로 생성한 계정은 동일한 id가 없는 경우에만 생성 된다.
Type	Primary
Cross Reference	Functions: R 1.2 Use Cases:
Pre-Requisites	N/A
Typical Courses of Events	(A) : Actor, (S) : System 1. (A) : 계정 생성을 요청한다. 2. (A) : ID/PW를 입력 한다. 3. (S) : ID/PW를 확인 하여 기존 사용자와 비교 한다. 4. (S) : 계정 생성에 적합한 경우 사용자 계정을 생성 한다.
Alternative Courses of Events	…
Exceptional Courses of Events	E1. 동일한 id를 가진 계정이 이미 존재하는 경우 생성하지 않는다.



Phase 2030. Analyze

2032. Refine Use case Diagram(유즈케이스 다이어그램 정제)

- Business use case Diagram을 검증하고 수정함
- 입력 : Business use case model, essential use case descriptions ***(Secure use case descriptions)***
- 출력 : A refined use case diagram
- 적용 표준 : UML의 use case diagram

- Step
  a. Essential use case diagram에 따라 Business use case diagram를 검토
  b. Use case 및 relationship을 추가 · 수정하여 Use case diagram을 정제

2033. Define Domain Model(도메인 모델 정의)

- Domain concept model을 정의
- 입력 : Business concept model, essential use case descriptions ***(Secure use case descriptions)***
- 출력 : A conceptual class diagram
- 적용 표준 : UML의 use case diagram

- Domain model이란
  - ✓ 현실에서 식별되는 정보를 토대로 만든 개념적 · 의미적 class들을 말함
  - ✓ 실제 software object 설계 시 기초 자료가 될 수 있음

Phase 2030. Analyze

2035. Define System Sequence Diagrams(시스템 순서 다이어그램 정의)

- Actor의 이벤트에 따른 system의 동작 순서 다이어그램 작성
- 입력 : essential use case 명세서, use case diagram
- 출력 : 시스템 순서 다이어그램(SSD)

- 시스템 순서 다이어그램(SSD) 이란
  - ✓ Actor의 행동에 따른 event 발생 과정을 시스템적으로 보여주는 그림
  - ✓ 모든 System은 black-box로 취급됨
  - ✓ SSD는 다음의 두 가지를 정의함
    - 주요 성공 시나리오 / 빈도, 복잡성, 대안 시나리오

- Step
  a. Use case에 기반하여 System으로 대표되는 것을 black box라 정함
  b. Use case에서 주요 event 과정을 보고 System에 직접적인 관련을 하는 actor들을 식별
  c. System boundary를 결정
     - Use case에서 각 actor를 통한 주요 event 발생 과정을 식별 (각 이벤트 호출을 하는 것을 system event라 함)
     - System event에 이름 지정***(보안 이벤트도 이 과정에서 추가)*** (동사+목적어 방식 이름을 지정함, ex: enterItem)
  d. 각 Actor에 할당된 SSD에 대한 use case text를 작성 (use case text는 전체 system event 과정에 대한 내용을 담음)

Phase 2030. Analyze

2036. Define Operation Contracts(동작 정보사항 정의)

- 시스템 운영에 필요한 각 동작의 정보사항(Contracts) 정의
- 입력 : System Sequence Diagram(SSD), conceptual class diagram
- 출력 : Operation Contracts

- 정보사항(Contract) 이란
  - ✓ 한 동작(operation), 즉 SSD에서 system event로 정의된 것들을 말하며 system operation이라고도 함.
  - ✓ 이것의 동작에 대한 여러 정보(실행목표, 타입, 예외사항, 출력, 선후조건 등)들을 표 등의 문서로 만든 것을 정보사항(contract)이라고 함

- Step
  a. System sequence diagram으로부터 system operation 식별
  b. System operation name을 contract name으로 함
  c. Contract에 여러 정보들을 기입함
     - ✓ Responsibilities, Type, Cross References, Notes, Exceptions, Output, Pre-Conditions, Post-conditions 등
       (Exception 부분에 검증내용 추가)
     - ✓ Pre-condition이란 해당 operation 실행 전 시스템 조건을 말함
     - ✓ Post-condition이란 해당 operation 실행 후 시스템 결과를 말함

Phase 2030. Analyze

2038. Refine System Test Case(시스템 테스트 케이스 정제)

- 추가 정보를 사용하여 system test plan을 수정 및 정제
- 입력 : essential use case description ***(Secure use case description),*** system test plan, 순서 다이어그램
- 출력 : 정제된 system test plan

- Step
  - ✓ analyze 프로세스 결과를 이용하여 1009번의 결과를 정제함
  - ✓ 입력값 검증을 위해 테스트 항목에서 실제 공격기법에 해당하는 모의해킹 테스트

Phase 2030. Analyze

Test Number	Test 항목	Description	Use Case	System Function
1	로그인 시험	Id, pw를 입력하고 로그인 시도를 하여 로그인 기능 test	1. Login	R 1.1
2	로그아웃 시험	로그아웃 버튼을 눌러 로그아웃 기능 test	2. Logout	R 1.1
3	계정 생성 시험	계정 생성 데이터를 입력하여 계정 생성 기능 test	3. Make Account	R 1.2
4	잔액 확인 시험	잔액 확인 버튼 동작 여부 test	4. Identify Balance	R 1.3
5	잔액 충전 시험	금액을 입력하고 잔액 충전 기능 test	5. Recharge Balanc e	R 1.4
6	인쇄 버튼 시험	인쇄 매수를 입력하고 인쇄 기능 test	6. Request Print	R 2.1
7	잔액 체크 시험		7. Check Balance	R 2.2
8	용지 확인 시험	용지 잔량 출력 test	8 Identify Paper	R 3.1
9	용지 충전 시험	용지 충전 test	9. Recharge Paper	R 3.2
10	사용자 목록 확인 시험	전체 사용자 목록 출력 test	10. Identify User	R 3.3
11	수익 확인 시험	총 수익 출력 test	11. Identify Money	R 3.4

1009번

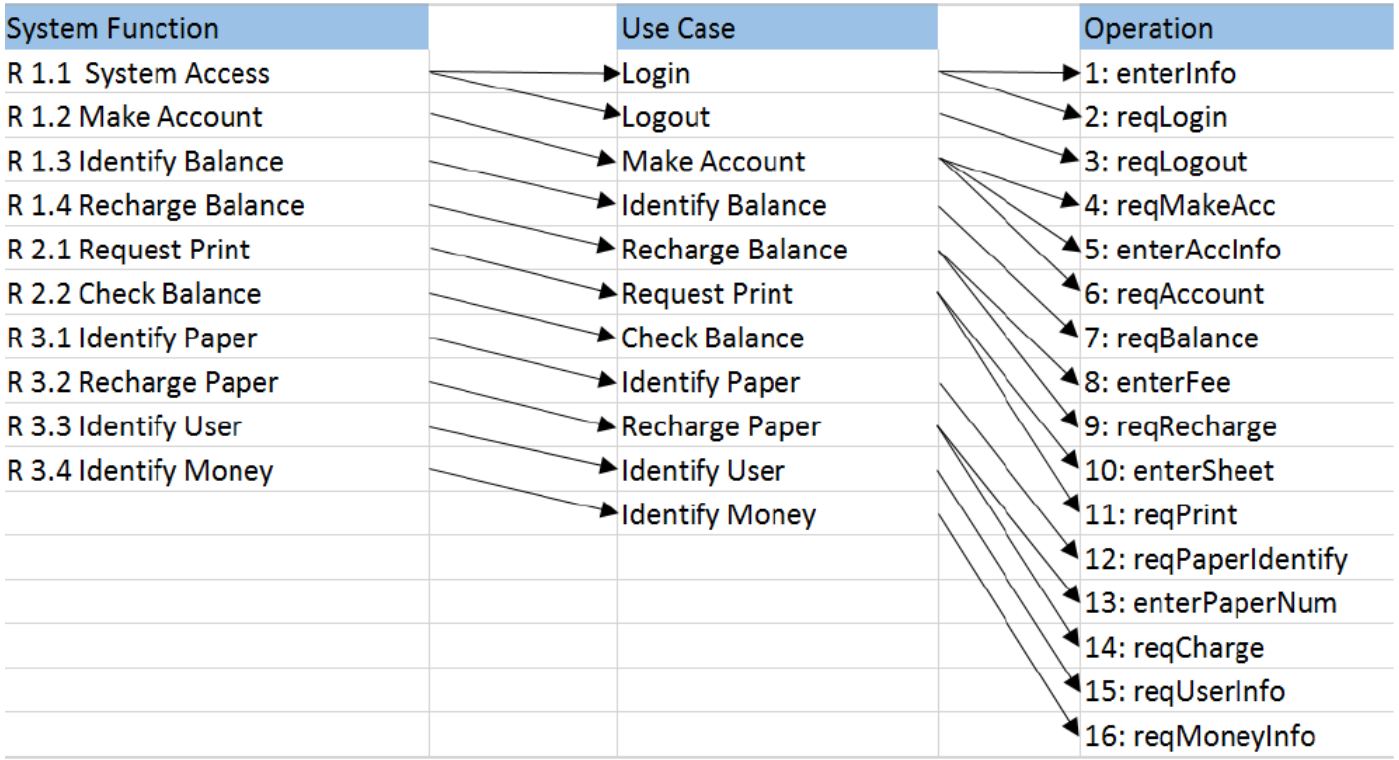
Test Number	Test 항목	Description	Use Case	System Function
1 – 1	로그인 시험	존재하는 계정의 Id, pw를 입력하고 로그인 시도를 하여 로그인 기능 test	1. Login	R 1.1
1 - 2	로그인 시험	존재하지 않는 계정의 Id, pw를 입력하고 로그인 시도를 하여 로그인 기능 test	1. Login	R 1.1
2	로그아웃 시험	로그인 상태에서 로그아웃 버튼을 눌러 로그아웃 기능 test	2. Logout	R 1.1
3	계정 생성 시험	계정 생성 데이터를 입력하여 계정 생성 기능 test	3. Make Account	R 1.2
4	잔액 확인 시험	잔액 확인 버튼 동작 여부 test	4. Identify Balance	R 1.3
5	잔액 충전 시험	금액을 입력하고 잔액 충전 기능 test	5. Recharge Balanc e	R 1.4
6	인쇄 버튼 시험	인쇄 매수를 입력하고 인쇄 기능 test	6. Request Print	R 2.1
7 – 1	잔액 체크 시험	잔액이 충분한 상태에서 인쇄 매수를 입력하고 인쇄 기능 test 수행	7. Check Balance	R 2.2
7 – 2	잔액 체크 시험	잔액이 부족한 상태에서 인쇄 매수를 입력하고 인쇄 기능 test 수행	7. Check Balance	R 2.2
8	용지 확인 시험	용지 잔량 출력 버튼을 통해 기능 test	8 Identify Paper	R 3.1
9	용지 충전 시험	충전할 용지 수량을 입력하고 용지 충전 test	9. Recharge Paper	R 3.2
10	사용자 목록 확인 시험	전체 사용자 목록 출력 test	10. Identify User	R 3.3
11	수익 확인 시험	총 수익 출력 test	11. Identify Money	R 3.4

2038번

Phase 2030. Analyze

2039. Analyze(2030) Traceability Analysis(Analyze 단계의 추적성 분석)

- 2030 단계의 결과물들의 연관성을 분석
- 입력 : Essential use case 명세서***(Secure use case description)***, sequence diagram, operation contracts
- 출력 : Traceability analysis 결과
- Steps : 각 단계의 결과물들의 관계를 직접 작성***(여기도 operation에 추가됨)***



Phase 2040. Design

2041. Design Real Use Cases

- 구체적 입출력 기술 및 전체적 구현 측면에서의 실제 use case 설계를 설명
- 만약 그래픽 UI가 연관되어 있으면, 실제 use case는 GUI diagram 및 하위 단계의 interface 위젯 간 상호작용을 포함해야 함
- 입력 : Essential Use Case Descriptions ***(Secure Use Case Descriptions)***
- 출력 : Real Use Case Descriptions

- Steps
  a. Essential use case에서 각 use case 식별
  b. 확장된 format에 UI 위젯을 추가시켜 주요 이벤트 과정의 구체적 구현 상세항목을 작성(UI widget에서 버튼 누르는 순서 등이 더 추가됨)

Phase 2040. Design

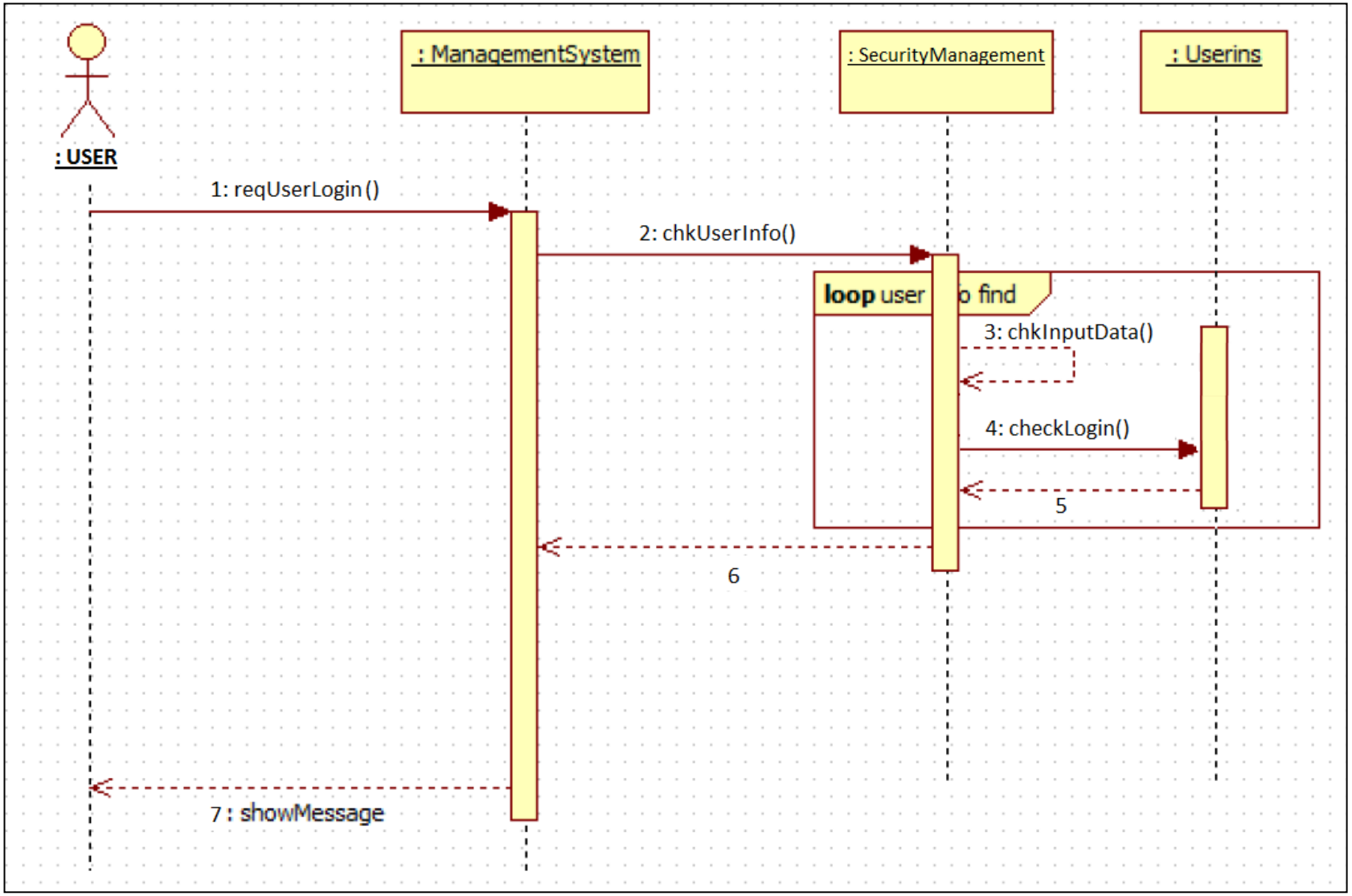
2044. Define Interaction Diagrams

- Object(Class)들이 어떻게 상호작용하는지를 나타내는 그림
- ***KUPE 예시에서는 Sequence Diagram으로 작성***
- 입력 : Real use case 명세서
- 출력 : interaction diagram
- 표준 적용 : UML's sequence diagram, Collaboration diagram
- Interaction diagram은 일반적으로 위의 2가지 특수 다이어그램으로 나뉨
  - ✓ Collaboration diagram : graph나 network format에서 object 간 상호작용을 표현
  - ✓ Sequence diagram : 일종의 경계 양식으로 표현되어 있으며, 각각의 object들이 우측으로 더해짐(일반적 시퀀스 다이어그램이 이런 형식)

- Steps
  a. Actor를 작성
  b. Real use case 명세서와 개념적 class diagram을 참고로 각 use case의 동작에 참여하는 object나 class를 배치
  c. Use case의 목표를 달성하기 위한 시스템의 object 간 상호동작을 설계 (모든 것의 시작은 use case 명세서로 시작)

Phase 2040. Design

➢ Login에 대한 sequence diagram



Phase 2050. Construct

2051. implement class & methods definitions(구현 클래스 및 메소드 정의)

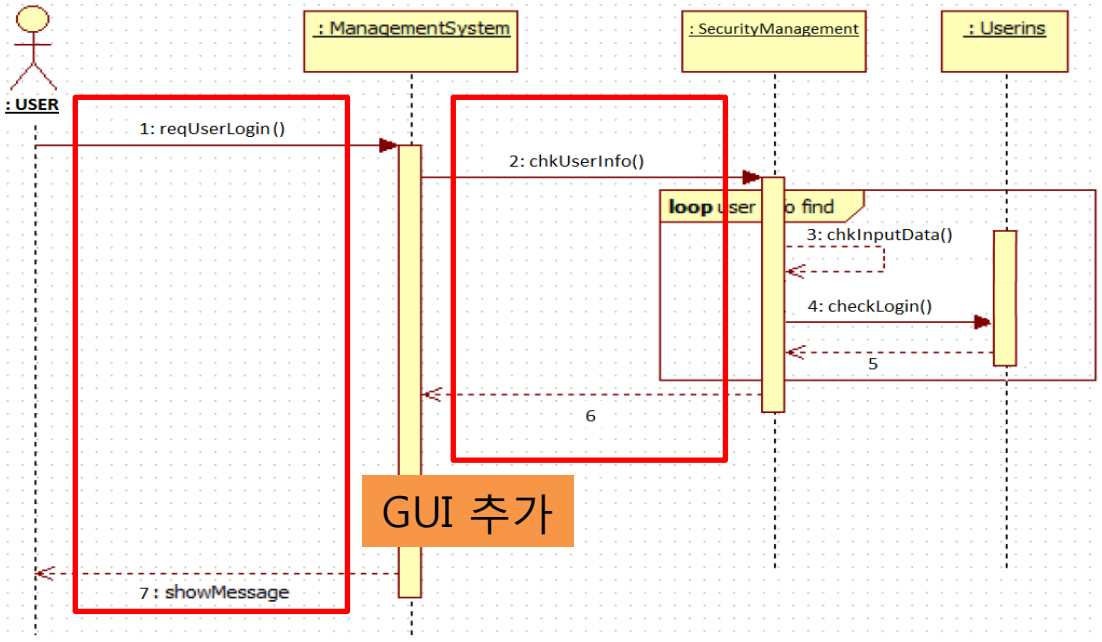
- Input : Design class diagram, real use cases, Interaction diagram
- Output : Class & Methods description
- Phase 2060의 'Unit test case' 설계에서 사용된다.
- Step
  a. Design class diagram(2045)의 모든 클래스와 메소드 식별
  b. 위 포맷을 사용함으로써 목표에 대한 설명 작성
  c. 구현

Type	Method
Name	chckinputData
Purpose	User가 ID를 입력하여 login을 확정하기 전에, 필터링을 실행하는 method
Cross Reference	Use Cases: 1. login
Input(Method)	N/A
Output(Method)	N/A
Abstract operation	N/A
Exceptional Courses of Events	N/A

Phase 2050. Construct

2052. implement windows (GUI와 오퍼레이션 사이 흐름정의)

- Input : Interaction diagram, design class diagram, real use case description, UI Story board, operation contracts
  Output : GUI implements results and description, Refined Class Diagram
- Step
  a. Actor와 System사이의 interaction diagram에서 GUI system operation
  b. GUI operation을 추가함으로써 개선된 interaction diagram 작성
  c. 다이어그램에서 GUI 오퍼레이션의 설명을 그림



reqUserLogin (GUI 입력폼)	(사용자 ID 및 비밀번호 입력 후 버튼클릭)	chkUserInfo 로 연결
chkUserInfo		Loop(chkInputData, checkLogin) 연결
showMessage	Message 출력	ShowMessage 로 연결

Phase 2050. Construct

Name	reqUserLogin
Responsibilities	Text box에 입력된 사용자 ID와 PW로 로그인을 요청한다.
Type	GUI
Cross References	N/A
Notes	Text box에 입력된 숫자를 화면에 표시한다.
Pre-Conditions	사용자ID와 PW가 입력되어야함
Post-Conditions	Login 버튼이 눌러져야 한다.

Phase 2050. Construct

2053. implement reports(분석정보, 개선된 디자인 및 프로그램 재구현된 디자인 결과에 대한 레포트 구현)

- Input : All of information
- Output : Analysis, design reports

※진단보고서 양식에서 취약점 설명위주로 작성

보안약점	유형	UAF(Use-After-Free) and Buffer Overflow, SQL Injection
	한글명	메모리 해제 후 참조 및 버퍼오버플로우, SQL 삽입

취약코드		1. UAF and Buffer Overflow
		`int main(int argc, char *argv[])`
		`{`
		`  char buffer[10];`
		`  if (argc <2)`
		`  {`
		`  fprintf(stderr, "%s Wn", argv[0]);`
		`  return 1;`
		`  }`
		`  strcpy(buffer, argv[1]);`
		`  return 0;`
		`}`
		2. SQL Injection
		`String tableName = props.getProperty("jdbc.tableName");`
		`String name = props.getProperty("jdbc.name");`
		`String query = "SELECT*FROM" + tableName + "WHERE Name =" + name;`
		`stmt =con.createStatement();`

취약사유		1. UAF and Buffer Overflow 특정 개체에 대한 메모리를 해제했는데도, 여전히 존재하는 객체를 참조하여 해제된 메모리 위치에 공격자가 원하는 값을 쓰거나 함수 포인터로 사용하여 시스템을 제어해 악성코드 및 프로그램을 자동실행하도록 함 2. SQL Injection 데이터베이스와 연동된 응용프로그램에서의 입력폼에 공격자가 SQL문을 삽입하여 DB로부터 데이터를 열람하거나 조작함.
보완조치방안		1. UAF and Buffer Overflow **①안전한 라이브러리의 함수 사용하여 메모리를 제한** 좀 더 개선된 함수 strlen s나 wcslen s 를 사용하여 참조할 최대 메모리를 제한 할 수 있다. 이 외에 기존의 문자열 함수 뒤에 _s가 붙은 이름의 함수를 사용하면 버퍼 오버플로 문제를 개선할 수 있다. ②포인터 보호 버퍼 오버플로를 방지하기 위한 접근 방법으로 스택이나 힙 상의 코드가 실행되는 것을 막는다. 공격자는 버퍼 오버플로를 이용하여 임의의 코드를 프로그램의 메모리에 삽입할 수 있지만, 실행 영역 보호가 있다면, 그 코드를 실행하고자 하는 어떠한 시도도 예외를 발생시킨다. ③주소 공간 배치 난수화 실행 코드의 기반 주소, 라이브러리, 힙, 스택 주소 등을 임의로 프로세서의 주소 공간에 배치한다. 2. SQL Injection **①입력값에 대한 파라미터를 길이를 제한**한 '상수 스트링'으로 생성 공격자가 DB에 접근하는 쿼리가 입력되지 않도록 입력값이 길이가 제한된 '상수 스트링'으로 설정한다. **②입력값이 쿼리의 구조를 변경시키는 것을 방지하도록 설정** 외부 입력값이 위치하는 부분을 '?'으로 설정하여 실행시 해당 파라미터가 전달되도록 수정하여 외부입력값이 쿼리의 구조를 변경시키는 것을 방지한다. **③입력값이 특정문구는 허용하지 않도록 필터링 설정** DB QUERY에 사용되는 외부입력값에 대하여 특수문자 및 특정쿼리를 제한하도록 필터링 설정.

출처:
http://www.openeg.co.kr/category/%EB%B3%B4%EC%95%88/%EC%8B%9C%ED%81%90%EC%96%B4%EC%BD%94%EB%94%A9?page=1

Phase 2050. Construct

2055. write unit test code

- Input : Implements results, class & methods definitions
- Output : Unit test code

- Example1) Buffer overflow unit test

```
public void testSetAllocate() {
        String cnt = request.getParameter("cnt");
        if(cnt == null)
        {
                cnt = "0";
        }
        int cntl = Integer.parseInt(cnt);
        String[] arr = new String[cntl];
        for( int i = 0; i<cntl; i++)
        {
                arr[i] = request.getParameter("r"+i);
        }
}
```

Phase 2050. Construct

2055. write unit test code

➢ Example2) SQL injection unit test code 작성

'	union
"	select
–	insert
#	drop
(	update
)	from
;	where
@	join
=	substr (oracle)
*	user_tables (oracle)
/	user_table_columns (oracle)
+	subsring (ms-sql)
	information_schema (mysql)
	sysobjects (ms-sql)
	table_schema (mysql)
	declare (ms-sql)

Regular Expression 사용하여
필터링 적용

```
public static string[] blackList = {"--",";--",";","/*","*/","@@","@",
                                    "char","nchar","varchar","nvarchar",

"alter","begin","cast","create","cursor","declare","delete","drop","end",
                                    "exec","execute","fetch","insert","kill","open",
                                    "select",  "sys","sysobjects","syscolumns",
                                    "table","update"};
```

Phase 2060. TEST

2063. System Testing

- Input : Implements results, system test plan and cases
- Output : System testing results, reports

Test Number	Test 항목	Description	Use Case	System Function
1-1	로그인 시험	존재하는 계정의Id, pw를 입력하고 로그인 시도를 하여 로그인 기능 test	1. Login	R 1.1
1-2	로그인 시험	존재하는 계정의Id, pw를 입력하고 로그인 시도를 하여 로그인 기능 test	1. Login	R. 1.1
1-3	로그인 시험	공격자가 주로 쓰는 삽입공격문을 입력하고 로그인 시도를 하여 로그인 기능 test	1. Login	R.1.1
2	로그아웃 시험	로그인 상태에서 로그아웃 버튼을 눌러 로그아웃 기능 test	2. Logout	R.1.1
3	계정생성 시험	계정 생성 데이터를 입력하여 계정 생성 기능test	3. Make Account	R.1.3


Q
&
A