Software Security

Security Development Process for KUPE

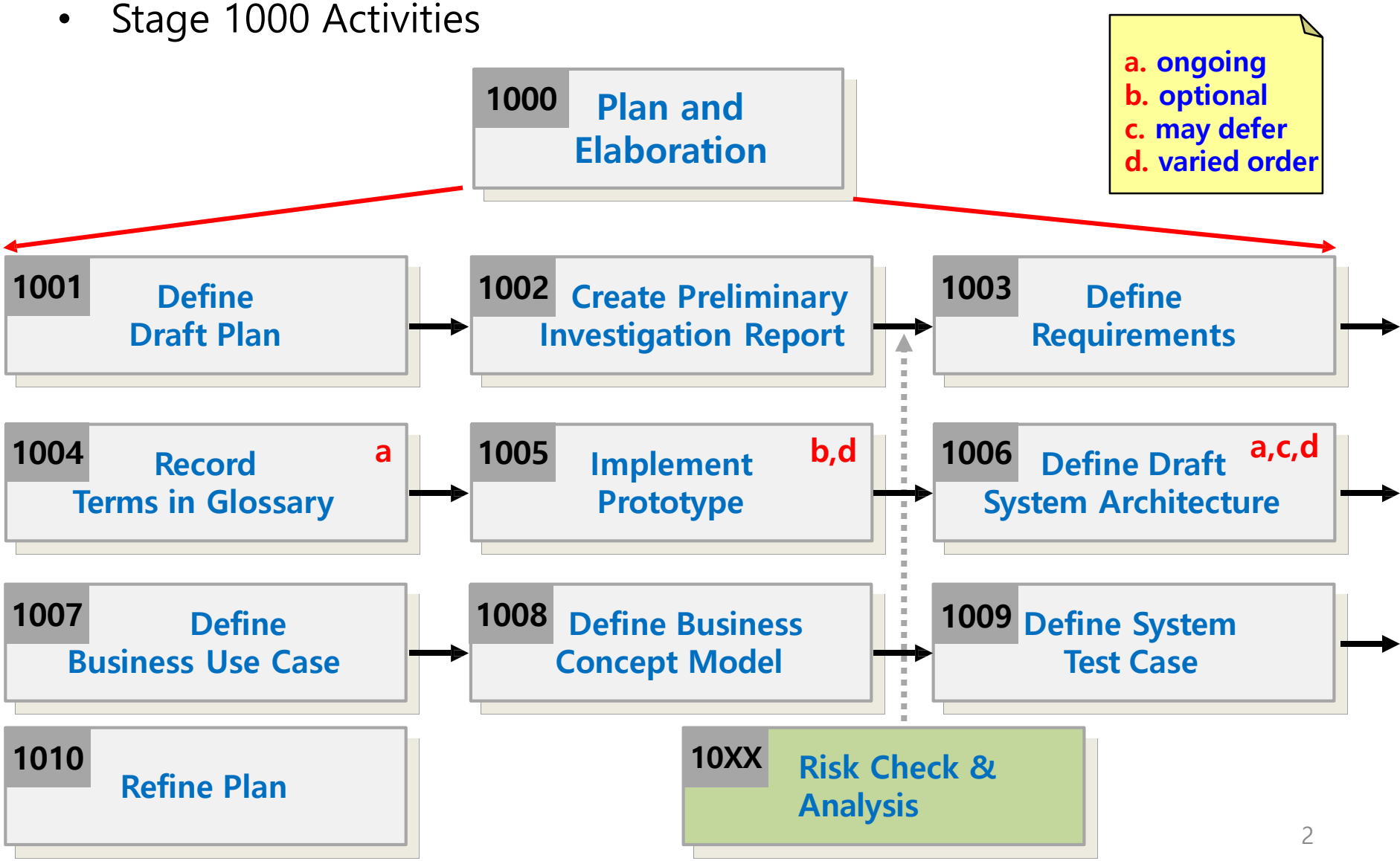
- 추가 본 -

IT융합 정보보호학과

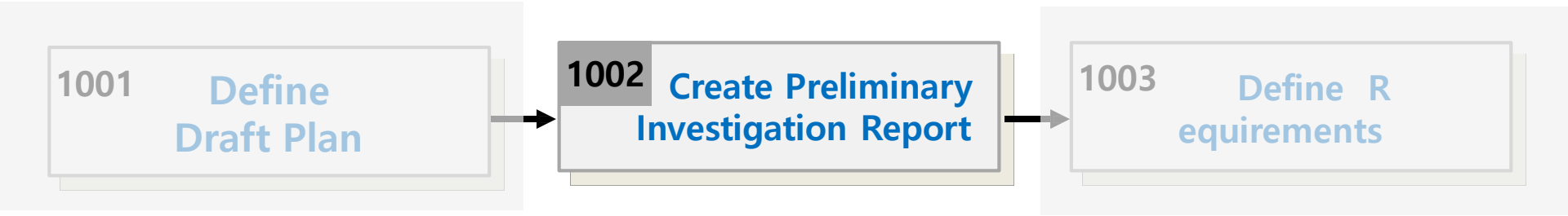
조 원 : 이 남 곤
안 정 현
발 표 자 : 두 상 균

Stage 1000. Plan and Elaboration

- Stage 1000 Activities



Activity 1002.
Create Preliminary Investigation Report



- Description
  - Write an investigation report on alternatives, business needs, **risk**, etc
  - Input : draft project plan risk
  - 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 10XX. Risk Check & Analysis

(Risk Report)- Ref. ISO/IEC 27000 series

- 위험 보고서(Risk Report)를 작성하기 위해 위험 요소에 대해 식별
  a. 자산 식별
     - 가지게 될 정보 자산(데이터)과 각 자산의 민감도 분석
  b. 위험 식별
     - 각 자산과 관련하여 발생할 수 있는 위험에 대한 식별
  c. 위험 분석
     - 식별 된 위험에 대해 어떠한 위험이며 정도에 대한 분석
  d. 위험 평가
     - 자산의 가치와 분석 된 위험의 정도를 분석하여 최종 평가
  e. 위험 처리
     - 평가된 위험의 정도에 따라 해당 위험의 처리 방법에 대해 결정

Phase 10XX. Risk Check & Analysis

(Risk Report)- Ref. ISO/IEC 27000 series

자산 식별

Function	Input Data	Sensitivity
Make Account	Privacy(Name, Addr, etc)	5
Recharge Balance	Business Cash	5
Check Balance	Balance	2

•

•

•

•

Phase 10XX. Risk Check & Analysis

(Risk Report)- Ref. ISO/IEC 27000 series

위험 식별

Function	Input Data	Vulnerability
Make Account	Privacy	SQL Injection
Recharge Balance	Business Cash	Buffer Overflow
Check Balance	Balance	Data Outflow

•

•

•

•

Phase 10XX. Risk Check & Analysis

(Risk Report)- Ref. ISO/IEC 27000 series

위험 분석

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

Phase 10XX. Risk Check & Analysis

(Risk Report)- Ref. ISO/IEC 27000 series

위험 평가 & 처리

Asset	Asset Valuation	Data Type	Risk	Risk Valuation	Cost	Treatment
Name	4	Privacy	SQL-Injection	2	2	Keep
Phone	5			5	2	Encryption
Addr	2			4	2	Acc Con

•

•

•

•

Activity 2030(sec). Check Possible 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) 시스템데이터 정보노출, **잘못된 세션에 의한 데이터 정보노출**

참조) 붙임1_소프트웨어_개발보안_가이드.pdf

Phase 2050. Construct

2055. write unit test code

➢ Example3) 중요정보 평문저장(b. 보안기능의 부적절 구현)

```
void foo() {
   try {
         Socket socket = new Socket("taranis", 4444);
         PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
         String password = getPassword();
         out.write(password);
   }
   catch (FileNotFoundException e)
   {
      ...
```

```
void foo() {
   try
   {
       Socket socket = new Socket("taranis", 4444);
       PrintStream out = new PrintStream(socket.getOutputStream(), true);
       Cipher c = Cipher.getInstance("AES/CBC/PKCS5Padding");
       String password = getPassword();
       encryptedStr= c.update(password.getBytes());
       out.write(encryptedStr,0,encryptedStr.length);
   }
   catch (FileNotFoundException e) {
```

Phase 2050. Construct

2055. write unit test code

➢ Example4) 하드코드된 패스워드(b. 보안기능의 부적절 구현)

```
Private static ENCRYPT_KEY="XXXXXXX...";
.....
SecretKeySpec skeySpec_de =
new SecretKeySpec
(ENCRYPT_KEY.getBytes(), "SEED");
```

암호화 키 값이 private하게 되어 있지만 Default값이 Static이므로 수정 불가 (노출 시 사용에 취약)

```
ENCRYPT_KEY=prop.getProperty("key");
ENCRYPT_KEY=decrypt(ENCRYPT_KEY);
.....
SecretKeySpec skeySpec_de =
new SecretKeySpec
(ENCRYPT_KEY.getBytes(), "SEED");
```

사용자가 입력한 값으로 암호화 키를 구성하는 동적 방식을 권장

Phase 2050. Construct

2055. write unit test code

➢ Example5) 오류 메시지를 통한 정보노출(c. 에러처리)

* Try-catch문 적용 시 default인 e.printStackTrace()

```
.....
} catch (IllegalAccessException e) {
    e.printStackTrace();
....
```

* Catch문의 처리방법 수정

```
.....
} catch (IllegalAccessException e) {
Log.error("Illegal Acess");
....
```

Phase 2050. Construct

2055. write unit test code

➢ Example6) 잘못된 세션에 의한 데이터정보 노출(d. 캡슐화)

```
Public class TEST1 extends HttpServlet
{
Private String name;
Protected void doPost (HttpServletRequest
req, HttpServletResponse res)
Throws ServletException, IOException
  {
Name = req.getParameter("name");
.....
Out.println(name + ", thanks for visiting!");
  }
}
```

```
Public class TEST1 extends HttpServlet
{
Protected void doPost (HttpServletRequest
req, HttpServletResponse res)
Throws ServletException, IOException
  {
String Name =
req.getParameter("name");
.....
If (name == null || "".equals(name))
Return;
Out.println(name + ", thanks for visiting!");
  }
}
```


Q
&
A