1. SQL Injection;
   SQL 인젝션이란 취약한 매개변수에 악의적인 코드를 넣는 공격이다.
   로그인 폼이나 검색창 등 직접적으로 입력을 받을수 있는 창으로부터 코드를 삽입하는 것 외에도, 직접적으로 입력을 받지는 않더라도 주소창등을 통해 입력받거나, 주고받는 신호를 직접 변조하는 방법도 있다.
   주로 코드 도중에 입력값을 받을때 SQL공격에 취약하다.

   직접적으로 코드를 짜넣기 위해 입력받는 구문을 ",',?>등으로 닫아버린 뒤, 악의적인 코드를 삽입하고 뒤쪽의 코드를 주석 등으로 없애버리는 방법을 사용한다.
   SQL인젝션을 방지하기 위해서는 ",',>,/,\등의 특수문자의 입력을 막고, 입력을 받는 변수를 한덩어리로 하지 않고 여러개로 나누거나, 입력 가능한 글자수를 제한하는등의 방법, 입력받는 값을 암호화 후 복호화 하는 방법 등이 있다.

   HTML, PHP, XSS 등 많은 언어에서 발생 가능하고, 주로 보안이 굉장히 취약한 사이트들에서 자주 발생하지만, 의외로 유명한 사이트들에서도 허술한 경우가 있다.

   SQL Injection 만으로도 데이터베이스의 값을 가져오거나 데이터베이스의 파일을 손상시키는 등의 악의적인 접근이 가능한만큼, 입력창을 만든다거나, 메인페이지로부터 직접 값을 받아오거나 주는 경우 주의해야만 한다.

2. 인증 결함
   인증 결함이란 로그인 등에 사용되는 인증에 필요한 계정정보를 노출하는 취약점이다. 소스코드 내부의 주석에 계정의 아이디 등이 적혀있다거나, 암호화된 어드민 계정 등이 있는 등이 대표적이다. GET방식을 사용하는 요청의 경우 주소창에 직접적으로 나타나는 경우도 있으므로 주의해야한다. 직접적으로 값이 노출되지 않더라도 Guessing 등으로 공격을 당할수 있으므로 가능한한 불필요한 정보의 노출을 피하는 편이 좋다.

   이러한 취약점이 발생하는 가장 큰 이유는 관리미흡이나 실수 등으로 벌어지는 만큼, 웹사이트를 제작하거나 유지보수 과정에서 이러한 실수가 없도록 해야하겠다.

3. XSS(크로스 사이트 스크립팅)
   서버에서 사용자에게 악성코드를 배포하도록 하는 공격 방식이다. 글이나, 그림(이미지파일)등 을 보여주는 위치에 SQL Injection을 활용하여 악성 코드를 넣는 단순한 방법부터 가짜 페이지를 만들어 악성코드를 배포하는 방법까지 다양하게 응용될 수 있다.

   크로스 사이트 스크립팅을 방지하기 위한 방법으로는 SQL Injection과 마찬가지로 특수문자 사용을 제약하거나, 입력구문이 종료되는 이후의 코드를 실행하지 않도록 하거나, 암호화를 시키는 등의 방법이 있다.

4. 취약한 객체 직접 참조
   중요한 객체(가격이나 수량 등)를 임의로 변경하여 서버에 요청하는 공격방법도 있다.
   중요한 객체를 사용자에게 직접적으로 노출시키는 것은 변조의 위험이 있으므로 지양하는 것이 좋다.

1. 민감데이터 노출
   데이터를 주고받는 과정에서 중요한 데이터를 암호화하지 않거나, 단순하게 암호화 시킬 경우 중간에 공격자가 정보를 탈취할 위험이 있다.
   sha1, base64등의 너무 단순한 암호화 방식은 사용을 지양하는것이 좋으며, 통신 주체사이에 자체적인 암호 키를 두어 중간에 탈취한 공격자가 암호 키가 없으면 해독하지 못하도록 하는 것도 좋은 방법이다.
   자체적인 암호화를 시키더라도 공격자가 중간에 통신자 사이에서 양쪽 모두와 암호화를 하고 주고받을 경우 중요 데이터를 탈취 당할 수 있다.(ARP Spoofing)

   데이터를 노출시키는 방법중 HeartBleed 라는 방식도 있다. openSSL에서 데이터를 요청할때 매 하트비트마다 최대 64kb의 메모리를 요청할 수 있는점을 이용하여, 실제 데이터를 초과하는 양의 데이터를 요청하여 서버 내부의 다른 데이터를 추가적으로 유출시키는 공격 방법이다.

2. 기능 수준의 접근 통제 누락
   서버나 페이지의 관리가 부실할 경우 공격자가 LFI,RFI등을 활용하여 서버에 파일이나 코드를 업로드하거나 삭제할 수 있다. 코드가 업로드 될 경우 사용자에게 코드를 배포시키는 등의 악용 또한 이루어질 수 있다.
   LFI. RFI를 막기 위해서는 사용되는 디렉터리 외부에서는 코드를 실행하지 않도록 하고, 특수문자 사용을 막는 등의 예방법이 있다.

   지원하는 디바이스마다 다른 페이지를 사용할 경우 각각의 페이지마다 권한이 다를 수 있으며, 특정 디바이스에서 접근했을때는 없었던 취약점이 다른 디바이스에서는 취약할 수 있다. 서버측에 디바이스의 정보를 변조해서 보내는 공격 방식도 있는만큼 주의가 필요하다.

   코드가 담긴 XML파일을 업로드하여 실행시켜 DDOS를 발하는 방식을 예시로 들 수 있다.

3. 크로스 사이트 요청 변조(CSRF)
   다른 사용자의 요청을 변조하여 악의적인 요청을 서버에 보내게 하는 방식이다. XSS와 같은 방법으로 코드를 실행시키며, 코드를 통해 서버에 요청을 보낸다는 차이점이 있다.

4. 알려진 취약점이 있는 컴포넌트 사용
   취약점이 알려졌지만 아직 패치가 나오지 않은 상태에서 진행되는 공격을 제로데이 공격이라고 한다. 패치가 이루어졌다고 하더라도 사용자나 서버가 업데이트 되지 않았다면 여전히 공격에 취약하다.
   PHP CGI Remote Code Execution이라는 취약점은 php의 CGI가 제대로 작동하지 않았던 취약점으로, 현재는 패치가 이루워졌지만 아직 업데이트를 하지 않은 장비들은 여전히 취약한 상태이다.
   셀쇼크 취약점은 환경변수를 설정할 때 설정코드 이후의 명령어를 그대로 실행했던 취약점으로, 현재는 환경변수를 처리하는 명령어 이후의 명령을 실행하지 않고 execute를 하는 방식으로 패치되었다. 마찬가지로 업데이트를 하지않은 장비는 취약하다.
   Buffer Overflow는 데이터의 크기를 이용한 취약점으로, 메모리에 입력받을 수 있는 데이터를 초과하는 데이터를 입력하여 다른 영역을 덮어씌워 장비를 다운시키거나, 공격자가 원하는 코드를 실행시키는 방법이다.
5. 검증되지 않은 리다이렉트 및 포워드
   공격자가 서버로부터 오는 신호를 조작하여 정해진 주소가 아닌 다른곳으로 이동하도록 만드는 공격방법이 있다. 공격자가 원하는 페이지로 이동시킬 수 있는만큼, 악성 사이트나 피싱사이트 등으로 이동하는 경우가 많다. 리다이렉션마다 쿠키를 바꿔주는 등의 방법으로 예방 가능하다.