Journal of KIIT. Vol. 22, No. 2, pp. 195-204, Feb. 28, 2024. pISSN 1598-8619, eISSN 2093-7571
http://dx.doi.org/10.14801/jkiit.2024.22.2.195

강화된 안전쿠키를 사용한 웹 응용의 보안에 관한 연구

정진호*, 차영욱**

A Study on the Security of Web Application using Enhanced Secure Cookie

Jin-Ho Jeong*, Young-Wook Cha**

이 논문은 안동대학교 기본 연구지원 사업에 의하여 연구되었음

요 약

쿠키와 세션은 HTTP의 무상태성을 극복하여 상태성을 생성하는 데 사용된다. 쿠키는 사용자의 데이터를 브라우저에 저장하고, 세션은 서버에 데이터를 저장하므로 쿠키보다 세션이 더 안전하다. 쿠키는 로그인과 같이 사용자 인증에 사용되며, 온라인 쇼핑몰의 장바구니와 같이 사용자가 저장하고 싶은 정보를 일시적으로 저장하는데 사용된다. 안전한 쿠키체계는 기밀성, 무결성, 인증 그리고 재생공격방지를 모두 만족해야 한다. 그러나 기존에 제시된 안전한 쿠키는 이 네 가지 요구 사항을 모두 충족하지 못하였다. 본 논문에서는 안전한 쿠키의 조건을 모두 만족하며, 세션보다 더 안전한 강화된 보안 쿠키를 제안한다. 또한 쿠키 값이 조작되면 인증을 할 수 없으며, 페이지가 이동할 때마다 쿠키 값이 지속적으로 변경되어 쿠키를 더욱 안전하게 보호할 수 있도록 설계되었다.

Abstract

Cookie and session are used to create statefulness by overcoming the statelessness of HTTP. Cookie stores users' data in the browser, while session stores the data in the server, so session is more secure than cookie. Cookie is used for user authentication, such as login, and are used to temporarily store information that users want to store, such as shopping carts in online shopping websites. Secure cookie must satisfy confidentiality, integrity, authentication, and anti-replay attack. However, the existing secure cookie methods do not satisfy all of these four requirements. In this paper, we propose enhanced secure cookie that satisfies all of the conditions of secure cookie, and more secure than session. Additionally, it is designed so that authentication cannot be performed when the cookie value is manipulated, and the cookie value is continuously changed whenever the page is moved to protect the cookie more secure.

Keywords
cookie, session, Anti-Replay, authentication, web security

* 디제이패밀리 대표
- ORCID: http://orcid.org/0000-0002-2602-3062
** 안동대학교 컴퓨터공학과 교수(교신저자)
- ORCID: http://orcid.org/0000-0002-9448-4899

· Received: Dec. 05, 2023, Revised: Jan. 11, 2024, Accepted: Jan. 14, 2024
· Corresponding Author: Young-Wook Cha
Andong National University, Korea
Tel.: +82-54-820-5714, Email: ywcha@anu.ac.kr

Ⅰ. 서 론

HTTP(Hypertext Transfer Protocol)는 웹과 그 외 다양한 애플리케이션에서 광범위하게 사용되어 왔고 지금도 사용중인 비상태성 프로토콜이다[1]. 그러나 실제 사용하는 웹 상에서는 사용자 로그인, 쇼핑몰에서 장바구니에 물건을 추가하여 저장하는 등 여러 방면에서 상태성이 요구된다. HTTP를 상태성으로 만드는 방법으로 쿠키와 세션이 제안되었다[2]. 사용자의 세션 데이터는 서버에 저장되며, 응답 헤더를 통해 세션ID가 브라우저에 보내져 쿠키에 저장된다[2]. 브라우저를 종료 시 생성된 쿠키가 사라지며, 세션파일은 일정시간 동안 서버에 존재한다[3]. 반면에 만료시간이 설정되는 쿠키는 데이터가 사용자의 브라우저에 저장되며, 브라우저가 종료된 후에도 쿠키 데이터는 사용자 컴퓨터에 존재한다[3]. 반면, 서버에 정보를 저장하는 세션은 저장되는 세션이 많을수록 서버에 부하가 증가한다[4]. 쿠키는 서버가 아닌 사용자의 컴퓨터에 정보를 저장하므로 세션에 비하여 서버에 부하는 주지 않는다. 한편 세션은 정보를 서버에 저장하므로 서버에 저장되는 데이터를 쿠키와 같이 사용자가 임의로 변조할 수 없으므로, 세션의 사용이 쿠키에 비하여 보안적인 측면에서 안전성이 높다. 이러한 특성을 가진 세션은 사용자 데이터의 기밀성, 무결성, 인증을 만족한다. 그러나, 세션ID가 노출되면 공격자는 브라우저를 통해 세션을 변경하여 사용자의 권한을 얻을 수 있으므로 재생공격방지에는 취약성이 있다.

본 논문에서는 서버에 부하를 줄 수 있는 세션을 사용하지 않고, 쿠키의 보안적인 측면을 보완하여 세션보다 안전하며 가용적인 측면에서 뛰어난 강화된 안전쿠키를 제안한다. 기존에 제시된 안전한 쿠키는[5][6] 기밀성, 무결성, 인증, 그리고 재생공격방지를 모두 만족하지는 못하였다. 본 논문에서 제안하는 강화된 안전쿠키는 사용자의 고유정보와 페이지 이동시마다 새롭게 생성되는 쿠키만료시간, 서버내에 숨겨져 있는 서버키를 합친 값으로 해쉬값을 만들어 쿠키로 사용하므로, 쿠키의 기밀성, 무결성, 인증, 재생공격 방지를 모두 만족한다. 지속적으로 변하는 해쉬값을 쿠키로 사용하므로, 쿠키의 탈취 공격에 안전하게 대처하며 쿠키에 저장된 개인정보도 보호된다.

Ⅱ. 관련 연구

2.1 세션과 쿠키의 보안위협

HTTP의 비상태성을 상태성으로 만드는 방법으로 제안된 쿠키와 세션은 클라이언트와 서버 간의 쿠키 전송을 통해 만들어진다[2]. 클라이언트인 사용자 브라우저와 웹서버 사이의 쿠키전송은 그림 1과 같다.

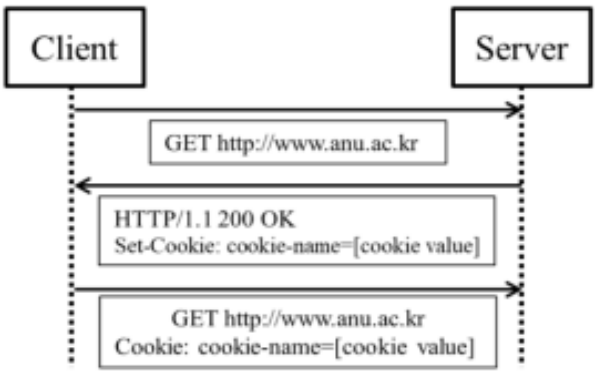

그림 1. 서버와 클라이언트 간 쿠키 전송
Fig. 1. Cookie transmission between server and client

서버에서 처음 생성된 쿠키는 HTTP 응답메시지의 Set-Cookie 헤더로 전달되어 브라우저에 저장된다[2]. 이때 쿠키는 클라이언트에서 저장이 필요한 데이터를 cookie-name 별로 분류하고 해당 데이터를 cookie value에 저장한다[2]. 세션은 cookie-name 대신 웹서버에 따른 세션 이름이 사용되고, cookie value 는 서버에서 생성된 session-id 가 저장된다[2].

쿠키와 세션은 사용한 브라우저의 쿠키 저장파일에 저장된다. 정상적인 세션은 브라우저를 종료할 시 쿠키 저장파일에서 사라지며, 쿠키는 브라우저를 종료한 후 만료시간이 지나더라도 저장파일에 남아있게 된다[3]. 웹서버의 세션 유지시간 동안 웹서버의 세션파일은 유지되므로, 세션이 유지되는 기간 안에 사용자는 자신의 session-id 값을 가지고 세션을 복구시킬 수 있다. 이는 session-id 값을 누군가 알고 있다면 다른 사용자 또한 해당 세션을 복구시킬 수 있음을 의미한다.

해당 기간 안에 세션을 복구하면 세션 유지시간은 다시 초기화된다.

세션과 쿠키는 모두 만료시간의 조작이 가능하다[7]. 세션의 만료시간을 조작하면 브라우저를 닫더라도 세션이 유지되며, 쿠키의 만료시간이 조작되면 조작된 만료시간까지 쿠키는 유지된다. 그러나 서버는 세션과 쿠키의 만료시간이 조작되었는지 알 수 없다. 만료시간이 조작된 세션과 쿠키는 브라우저 종료 후 만료시간이 지나더라도 쿠키 저장파일에 남아있다. 그러나 세션의 경우 웹서버의 세션 유지시간이 지나면 session-id 값을 가지고 세션을 복구하더라도 복구되지 않으며, session-id 값 자체에는 사용자 데이터를 포함하지 않는다. 반면에 쿠키는 저장된 쿠키값을 통해 재생이 가능하며 만료시간을 조작하여 쿠키의 유지가 지속적으로 가능하다. 또한 쿠키값 자체에 사용자 정보가 내포되어 있으므로, 쿠키값이 암호화되어 있지 않으면 쿠키값을 통하여 사용자 정보를 알 수 있게 된다. 이런 측면에서 세션이 쿠키보다 안전하다고 볼 수 있다.

세션은 서버에 세션파일로 저장된다. 세션 파일의 최대 크기는 웹서버의 설정에 따라 다르지만, PHP 기준 세션 파일 최대 크기의 기본값은 128MB이다[8]. 저장되는 세션 데이터가 많아질수록 파일의 크기는 커지게 되며, 파일을 불러오는 속도는 저하되므로 서버에 부하를 주게 된다. 쿠키는 사용자의 브라우저에 값을 저장하고, 최대 크기는 브라우저에 따라 약간의 차이가 있으나 구글 크롬 브라우저 기준 쿠키당 4096 bytes이다[9]. 쿠키는 최대 크키가 세션보다 크지 않고 데이터를 브라우저에 보관하기 때문에 세션에 비하여 서버에 부하를 주지 않는다. 서버에 부하를 주는 세션을 사용하지 않고 보안적으로 안전한 쿠키 사용을 위해 제안된 안전한 쿠키체계는 쿠키의 기밀성, 무결성, 인증, 재생공격방지를 모두 만족하는 것을 의미한다[10].

쿠키에는 사용자의 민감한 정보가 포함될 수 있으며, 정보가 노출될 시 개인정보 유출로 이어질 수 있다. 쿠키 기밀성은 클라이언트에 저장된 쿠키정보를 서버를 제외한 다른 클라이언트와 공격자가 읽지 못하게 하는 것을 의미한다[11]. 무결성은 특정 데이터가 임의로 생성되거나 삭제 혹은 변조되지 않도록 보호하는 것을 의미한다. 쿠키 값에 사용자의 로그인 정보가 있는 경우 쿠키를 변조하여 다른 사용자의 정보로 로그인이 가능할 수도 있다. 또한 실제 존재하지 않는 상품을 변조하여 쇼핑몰의 장바구니 쿠키에 집어넣는다면, 서버의 운영과 보안에 치명적인 영향을 줄 수 있다[12].

사용자가 아이디와 패스워드로 로그인 시도 시에 합당한 사용자로 판단되면 서버는 인증 쿠키를 만들어 사용자에게 전송한다. 브라우저는 서버와 통신 시 쿠키를 전달하여 해당 쿠키가 유효한지 검증받는다[13]. XSS(Cross-Site Scripting)는 OWASP TOP 10 2021부터 3번째 위협요소인 Injection의 한 부분이 된 공격이다[14]. XXS 공격으로 사용자의 쿠키나 세션을 탈취한 공격자는 해당 사용자의 쿠키나 세션을 재생하여 로그인하는 공격에 사용된다. XSS 공격뿐만 아니라 공공 PC 혹은 타 PC의 브라우저에 저장된 쿠키를 탈취할 수 있다. 공격자가 탈취한 쿠키를 재생하여 인증시도 시, 유효하지 않음을 감지할 수 있어야 재생공격을 방지할 수 있다[5].

2.2 안전한 쿠키체계

안전한 쿠키(Secure cookie) 체계에서는 쿠키의 기밀성, 무결성, 인증, 그리고 재생공격방지를 모두 만족할 수 있어야 하지만, 기존의 안전한 쿠키체계[5][6]는 기밀성이나 재생공격방지 같은 한 두가지 항목을 충분히 제공하지 못하고 있다. 웹 프로그램 통계제공 웹사이트, W3Techs의 2023년 7월 통계에 따르면, Wordpress는 전체 CMS(Content Management System) 시장의 63.1%를 점유하며, 전체 웹사이트의 43.2%를 차지하고 있다[15]. Wordpress의 로그인은 A. X. Liu가 제안한 안전한 쿠키방식을 사용하고 있다[5][16]. Liu의 안전한 쿠키체계는 그림 2와 같다.

$$\text{user name}|\text{expiration time}|(\text{data})_k$$
$$|\text{HMAC}(\text{user name}|\text{expiration time}|\text{data}|\text{session ID}, k)$$
where k=HMAC(user name|expiration time, sk)

그림 2. Liu의 쿠키[5]
Fig. 2. Liu's cookie[5]

사용자 이름과 쿠키의 만료시간, 키값 k로 암호화한 데이터, 그리고 SSL(Secure Sockets Layer) Session ID 등을 k로 해싱한 결과를 서로 붙여서 쿠키를 생성한다. Liu 쿠키의 문제점은 사용자의 이름이 암호화되지 않기 때문에, 저장된 쿠키를 통하여 해당 클라이언트가 어떤 사용자로 인증이 되었는지 알 수 있게 된다. 이는 쿠키의 기밀성이 만족되지 않는다고 볼 수 있다.

Liu의 쿠키는 재생공격방지를 위해 SSL Session ID를 사용하였다. 웹 프로그램 통계제공 웹사이트인 W3Techs의 2023년 7월 통계에 따르면, 전세계 약 16.7%의 웹사이트가 아직 HTTPS가 아닌 HTTP를 이용하고 있다[17]. HTTP만 사용하는 웹사이트에서는 SSL이 사용되지 않으므로, Liu 쿠키의 재생공격방지 기능이 작동되지 않는다. 대부분의 백엔드 언어에서는 SSL Session ID 값을 가져오지 못한다. 따라서 SSL Session ID 값을 지원하지 못하는 PHP나 ASP 같은 백엔드 언어에서는 재생공격방지가 이루어질 수 없다. 이에 따라 Wordpress 에서도 Liu의 쿠키를 사용 시 SSL Session ID를 제외하여 사용하고 있다[5][16]. SSL Session ID를 제외하고 사용하면 Liu의 쿠키는 data가 갱신되지 않는 한 쿠키는 고정값이 된다. 따라서 다른 사용자가 해당 쿠키값을 가질 수 있으면 해당 쿠키값으로 쿠키를 재생하고 쿠키의 만료시간을 조작하여 다른 사용자 명의로 서비스 이용이 가능하게 된다.

Liu 쿠키체계의 기밀성을 보완한 W. B. Lee et al.[6]의 쿠키체계는 그림 3과 같다. 이 방법은 일반적인 데이터와 개인정보에 민감한 데이터를 먼저 구분하는 것으로 시작한다. 큰 소수의 선택과 계산을 통해 키를 생성하여 일반적인 데이터에 붙인 후 해쉬함수로 해쉬값을 생성하여 서버키를 만든다. 민감한 데이터는 생성된 서버키를 사용하여 대칭키로 암호화된 데이터를 생성하고, 일반적인 데이터와 암호화된 데이터를 합쳐 쿠키를 만든다. 생성된 쿠키는 사용자 브라우저에 저장하고, 서버는 사용자 브라우저에 저장된 쿠키를 가지고 앞서 선택한 소수와 계산을 통하여 생성한 키를 역으로 계산하여 찾는다. 발견한 키를 통하여 암호화 했던 데이터를 복호화하였을 때 복호화된 데이터가 인식 가능한 데이터면 유효하고, 그렇지 않으면 인증이 거절된다.

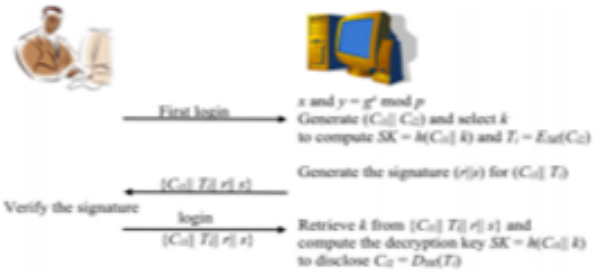

그림 3. Lee의 쿠키[6]
Fig. 3. Lee's cookie[6]

Lee[6] 쿠키의 문제점은 사용자별 인증 시 생성되는 쿠키를 사용자가 재인증하지 않거나 쿠키 데이터가 갱신되지 않는 경우에 항상 같은 값으로 존재한다는 것이다. 따라서 해당 쿠키가 공격자로부터 탈취되는 경우에 외부에서 인증할 수 있는 재생공격에 여전히 취약점이 존재한다. 공격자가 쿠키를 재생하여 사용하더라도 쿠키값 자체는 변조되지 않았기 때문에 유효성을 검증하는데 전혀 문제가 없다. 따라서 Liu의 쿠키와 마찬가지로 재생공격에 취약하다.

III. 강화된 안전쿠키 체계와 안전성

3.1 강화된 안전쿠키 체계

본 논문에서는 안전한 쿠키체계의 조건인 기밀성, 무결성, 인증, 그리고 재생공격방지를 모두 만족하는 강화된 안전쿠키(ESCK, Enhanced Secure CooKie)를 제안한다. 강화된 안전쿠키는 그림 4와 같이 표현된다.

Hash(Username|Cookie's new Expiration|Secret Key)|(Data|Secret Key)k
where k = Hash(Username|Cookie's new Expiration|Secret Key)||Secret Key

* | is separator * || is merging string

그림 4. 강화된 안전쿠키
Fig. 4. Enhanced secure cookie

먼저 회원의 아이디 혹은 고유정보, 현재 시간에 짧은 시간을 더한 쿠키의 만료시간, 그리고 서버 비밀키(Secret key)를 구분자 | 로 연결하여 해쉬값을 생성한다. 암호화키 k는 앞서 만들어진 해쉬값의 뒤에 서버 비밀키를 붙여 생성한다.

Data의 앞이나 뒤에 구분자와 함께 서버 비밀키

를 추가한 후 k로 대칭키 암호화한다. 해쉬값과 암호화된 값을 구분자 | 로 연결하여 생성된 결과가 본 논문에서 제시하는 강화된 안전쿠키이다. 웹서버는 쿠키값 중에서 구분자 | 의 왼쪽에 있는 해쉬값과 쿠키의 만료시간을 데이터베이스 회원테이블에 특정 컬럼으로 저장한다. 회원이 다른 페이지로 이동하거나 페이지의 새로고침 시에 만료시간을 현재시간에 짧은 시간을 추가하여 그림 4와 같이 강화된 안전쿠키를 다시 생성한다. 그 후 새롭게 생성된 구분자 | 의 왼쪽에 있는 해쉬값과 쿠키의 만료시간을 데이터베이스에 갱신시킨다.

3.2 강화된 안전쿠키의 기밀성과 무결성

3.2.1 기밀성

강화된 안전쿠키의 생성에 사용되는 서버 비밀키는 서버 내에 하드코딩 되어 비밀키로 사용되는 값이다. 그림 4에서 강화된 안전쿠키의 해쉬값 생성에 사용된 회원의 고유정보와 쿠키의 만료시간은 공격자가 알 수도 있는 정보이며, 공격자는 이들 정보를 조합하여 해쉬값을 생성해 봄으로써 크래킹 시도를 해볼 수 있다. 하지만 여기에 서버 비밀키를 포함함으로써 공격자의 크래킹이 거의 불가능하도록 하였다. 암호화가 필요한 데이터가 있는 경우 공격자는 암호화된 Data값의 암호화 키값이 구분자 | 의 앞에 있는 해쉬값이라고 추측할 수도 있다. 해쉬값에 비밀키를 한번 더 포함하여 Data의 암호화 키값을 예측할 수 없게 하였다. 안전한 해쉬함수는 일방향 특징을 가지고 있어 복호화할 수 없다[18]. 이에 따라 이 해쉬값과 서버 비밀키를 키로 한 Data의 대칭키 암호화 결과값도 키 값을 알 수 없으므로 쿠키의 기밀성이 보장된다.

3.2.2 무결성

어떤값의 무결성 체크를 위하여 일반적으로 해쉬함수로 생성되는 해쉬값을 사용한다. 이는 해쉬함수의 특성 중 입력값이 다르면 해쉬값도 항상 다르다는 충돌저항성과 관련있다[19]. 강화된 안전쿠키의 해쉬값은 회원의 고유정보와 쿠키의 만료시간, 그리고 서버 비밀키를 합하여 생성된다. 그리고 Data와 서버 비밀키를 합하여 대칭키 암호화한 결과를 앞서 만든 해쉬값과 구분자 | 로 연결한다. 쿠키가 어떤 값으로 이루어져 있는지 알 수 없는 공격자는 해쉬함수의 충돌저항성에 따라 임의의 값으로 동일한 쿠키를 생성 및 인증할 수 없게 된다.

쿠키는 조작이 가능하므로 공격자가 Data 부분의 쿠키를 조작할 시, 시스템에 따라 부하를 야기하거나 Data가 변조될 위험이 있다. 완성된 쿠키값에 | 와 같은 구분자가 존재하는지 확인하여 쿠키가 변조되었는지 확인할 수 있다. 또한 Data의 복호화 시에 Data 앞이나 뒤에 붙였던 서버 비밀키만 확인하여 복호화된 값을 검증할 수 있다면 쿠키 조작의 위험으로부터 안전할 수 있다. 예를 들어 서버 비밀키를 “ANU13579” 라고 가정할 시, 공격자가 암호화된 Data 값을 조작한다면, 시스템은 조작된 암호값을 키로 복호화하여 Data의 앞이나 뒤에 | 를 포함한 “ANU13579”가 있는지 판단한다. “ANU13579”가 있다면 올바른 Data 값으로 처리하고, 없다면 쿠키를 초기화시켜 공격자의 공격으로부터 안전할 수 있다.

3.3 강화된 안전쿠키의 로그인 인증 및 쿠키의 유효성 체크

웹 애플리케이션에서의 로그인 인증과 인증 과정에서 생성된 강화된 안전쿠키의 유효성 체크절차는 그림 5와 같다.

사용자가 아이디와 패스워드를 입력하면 해당 사용자 정보의 패스워드가 일치하는지 서버가 판단한다. 정보가 일치하면 쿠키를 생성하고 데이터베이스에 생성된 해쉬값과 쿠키의 만료시간을 저장한다. 생성된 쿠키는 사용자의 아이디, 쿠키의 만료시간으로 이루어져 있으므로 확실하게 고유한 값이다. 이에 따라 사용자는 브라우저에 저장된 쿠키로 데이터베이스 내 회원테이블의 해쉬 컬럼을 검색하여 사용자 정보를 가져올 수 있다. 사용자의 페이지 이동이나 페이지의 새로고침 시에 서버 측에서 쿠키의 검증과 만료시간을 체크한다. 쿠키가 조작되지 않았고 만료시간이 남아 있다면, 강화된 안전쿠키의 확인절차를 통하여 TRUE가 리턴되며, 인증의 유지를 위해서 사용자 브라우저와 데이터베이스에 저장된 해쉬값과 쿠키의 만료시간을 갱신한다.

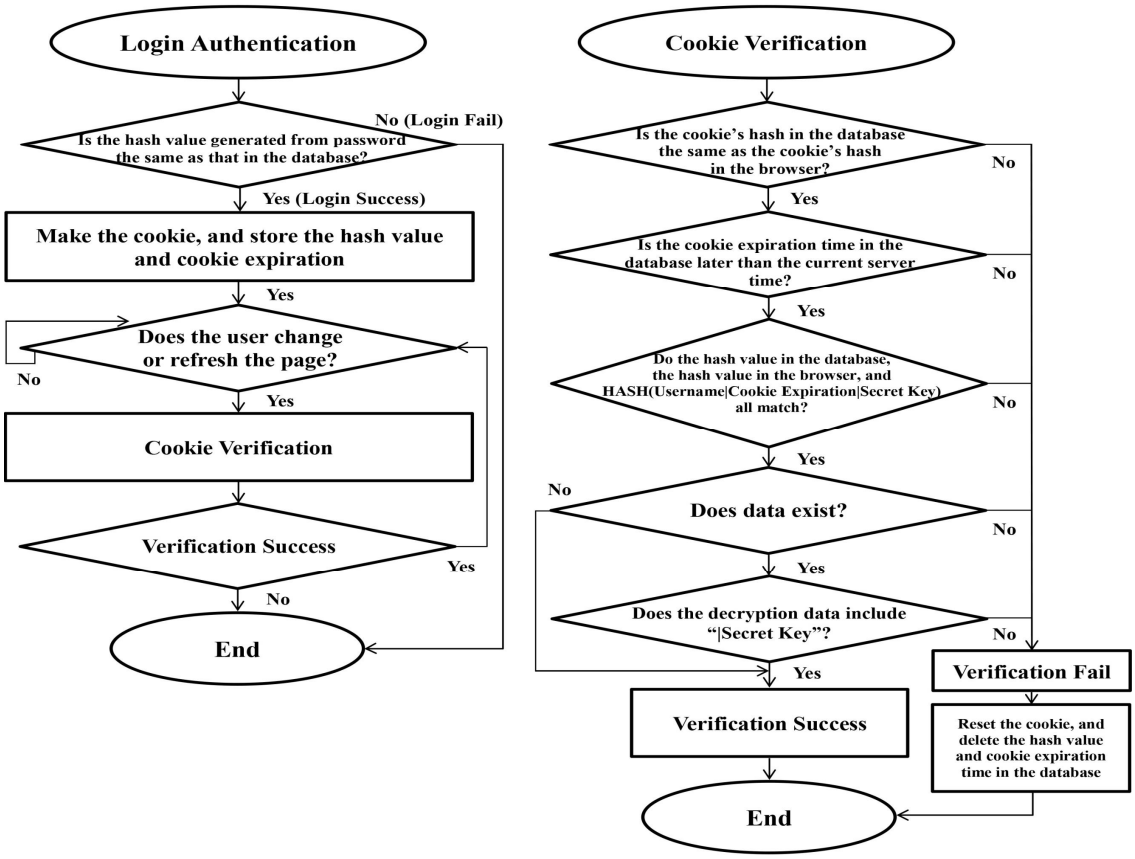

그림 5. 로그인 인증 및 강화된 안전쿠키의 유효성 체크절차
Fig. 5. Login authentication and enhanced secure cookie verification procedure

만약 쿠키 검증에 실패하여 확인절차에서 FALSE가 리턴되거나 사용자가 로그아웃 버튼을 누르면, 브라우저에 저장된 쿠키 및 데이터베이스에 저장된 해쉬값과 쿠키의 만료시간을 삭제한다. 쿠키 해쉬값과 만료시간이 데이터베이스에서 제거되면, 사용자의 쿠키값이 탈취되어도 공격자에 의한 인증이 불가능하다. 데이터베이스의 해쉬값, 브라우저에 저장된 쿠키값, 그리고 현재 사용자가 인증하여 사용중인 사용자 이름, 데이터베이스에 저장된 쿠키 만료시간, 서버 비밀키를 가지고 만든 해쉬값까지 이 3가지 값들을 모두 비교하면, 사용자 브라우저의 쿠키값 조작뿐만 아니라 데이터베이스의 조작까지도 감지 및 차단할 수 있어 사용자 정보를 보다 안전하게 보호할 수 있다. 데이터베이스에 해쉬값과 쿠키의 만료시간이 없으면 사용자가 가지고 있는 쿠키를 검증할 수 없으므로, 강화된 안전쿠키 방식에서 조작을 통한 로그인은 불가능하다. 웹사이트 관리자는 사용자의 쿠키와 만료시간을 강제로 삭제함으로써, 로그인된 사용자를 강제로 로그아웃 시킬 수 있어 회원 인증을 강화시킬 수 있다. 또한 데이터베이스에 저장된 쿠키의 만료시간이 남아있는 사용자들을 검색하여 현재 접속중인 사용자들을 파악할 수 있어 회원관리에도 도움을 줄 수 있다.

3.4 강화된 안전쿠키의 재생공격방지

강화된 안전쿠키의 구성요소 중 쿠키의 해쉬값과 만료시간은 사용자 로그인 시에 값으로 저장된다. 만약 공격자가 강화된 안전쿠키의 비밀키와 생성방법을 알더라도 데이터베이스에 해당 값이 없으면 인증이 불가능하게 된다. 쿠키의 만료시간은 사용자가 페이지를 이동하거나 페이지가 새로고침될 때마다 현재 시간에 짧은 시간을 추가하여 새로 갱신한다. 이에 따라 사용자가 페이지를 이동하거나 새로고침될 시에 만료시간이 변하므로 쿠키가 지속적으로 변하게 된다. 쿠키의 재생공격은 공격자가 해당 쿠키를 만료시간 안에 가져와 자신의 브라우저에서 작동 시킬 수 있어야 한다.

공격자가 짧은 쿠키 만료시간 안에 사용자의 쿠키를 가져오지 못하면 쿠키의 재생공격은 불가능하다. 사용자가 페이지 이동을 하거나 페이지 새로고침을 한다면 쿠키가 변하게 되어 재생공격을 할 수 없게 된다. 만약 사용자가 로그아웃 버튼을 누르지 않고 브라우저를 끄게 되어 쿠키가 변하지 않더라도, 만료시간이 짧기 때문에 재생공격이 쉽지는 않다. 공격자가 쿠키의 만료시간을 조작하여 만료시간을 강제로 늘린다고 하여도 데이터베이스에 만료시간이 저장되어 있어 인증을 차단할 수 있다.

현재 시간에서 짧은 시간을 추가하여 쿠키의 만료시간이 결정되며, 짧은 시간은 웹사이트의 컨텐츠에 의해 좌우된다. C. Liu et al.[20]의 연구에 따르면, 평균적으로 웹페이지에 머무는 시간은 1분 이내이다. 하지만 유튜브 같은 비디오 스트리밍 컨텐츠의 웹사이트에서 1분은 매우 짧은 시간일 수 있다. EU의 ePrivacy Directive에서는 쿠키 지속시간이 12개월이 넘으면 안된다고 한다[21].

해당 웹사이트에서 사용자가 한 페이지에 머무는 시간을 예측하여 만료시간에 적용되는 짧은 시간을 정할 수 있다. 로그인한 사용자가 해당 쿠키의 만료시간 전에 자동으로 새로고침 시킨다면, 페이지를 켜놓고 다른 일을 하다가 만료시간이 지나서 로그아웃 되는 것을 막을 수 있을 것이다. PHP.net에 따르면 세션 만료시간(session.gc_maxlifetime)의 기본값은 1440으로 24분이다[22]. 쿠키 만료시간을 세션의 만료시간보다 짧게 하면, 브라우저를 강제로 닫은 후의 인증 값 유지 측면에서 세션보다 보안적으로 나은 효과를 얻게 된다.

Ⅳ. 강화된 안전쿠키와 기존 쿠키의 비교

4.1 강화된 안전쿠키와 기존 쿠키의 안전성 및 호환성 비교

표 1은 안전한 쿠키체계 요소들에 대하여 기존 쿠키방식과 본 논문에서 제안하는 강화된 안전쿠키에서의 지원 여부를 나타낸다. 본 논문에서 제안하는 강화된 안전쿠키는 안전한 쿠키체계 요소인 기밀성, 무결성, 인증, 그리고 재생공격방지를 모두 지원한다. 반면, Liu의 쿠키는 기밀성과 재생공격방지를 지원하지 못하며[5], Lee의 쿠키는 재생공격방지를 지원하지 못한다[6]. Liu의 쿠키는 SSL이 지원하지 않는 환경에서는 사용할 수 없으므로 호환성에 문제가 있지만, Lee의 쿠키와 본 논문에서 제안한 강화된 안전쿠키는 모든 환경에서 사용이 가능하다.

표 1. 기존 쿠키와 강화된 안전쿠키의 비교
Table 1. Comparison of existing cookies and enhanced secure cookie

Elements of secure cookie scheme	Liu	Lee	Enhanced secure cookie
Confidentiality	X	O	O
Integrity	O	O	O
Authentication	O	O	O
Anti-Replay attack	X	X	O
Compatibility	△	O	O

기존 쿠키방식은 공격자에게 쿠키의 생성 방법과 비밀키가 노출되는 경우, 공격자는 공격자가 인증을 원하는 사용자 이름과 쿠키만료시간으로 자유롭게 인증을 위한 쿠키값 생성이 가능하다. 반면 강화된 안전쿠키는 데이터베이스에 쿠키값과 만료시간이 빈 값으로 있을 경우 쿠키 인증이 불가능하다. 사용자가 로그아웃을 누르지 않고 강제로 브라우저를 종료하여 쿠키값이 데이터베이스에 남아있더라도 짧은 쿠키 만료시간으로 인하여 인증이 쉽지 않게 된다.

4.2 강화된 안전쿠키와 기존 쿠키의 성능 비교

표 2는 본 논문에서 제시한 강화된 안전쿠키 방식의 쿠키 생성시간과 로그인 인증시간, 페이지 이동 시의 쿠키 검증시간을 Liu 쿠키보다 보안에서 우수한 Lee[6]의 쿠키방식과 비교한 결과이다. 각 시험의 소요시간은 시험 당 100회 측정하여 평균한 값으로 기재하였다. 해쉬 알고리즘은 SHA-256, 그리고 대칭키 알고리즘은 AES-256을 사용하였다. 시험을 위한 코드는 PHP 7.3으로 구현하였으며, MySQL 8버전, 3.40GHz CPU, 그리고 DDR3 16GB 메모리 환경에서 시험하였다.

표 2. 기존 쿠키와 강화된 안전쿠키의 성능 비교
Table 2. Performance comparison of existing cookies and enhanced secure cookie

Parameters of performance	Lee	Enhanced secure cookie
Create cookie value	0.026 ms	0.024 ms
Login authentication	12.668 ms	12.596 ms
Cookie validation when moving pages	12.659 ms	21.435 ms

임의의 데이터를 통한 쿠키값의 생성에는 Lee의 쿠키방식이 약 0.026ms, 강화된 안전쿠키 방식이 약 0.024ms 의 시간으로 측정되었다. 측정된 시간은 소수점 넷째 자리부터 버림하였다. 3가지 쿠키 방식 모두 매우 근소한 차이로 거의 시간차가 없다고 볼 수 있다. 로그인 인증에서는 Liu와 강화된 안전쿠키가 각각 12.668ms, 12.596ms의 시간으로 측정되었다.

3가지 쿠키방식이 쿠키값 생성과 다르게 로그인 인증에서 시간이 증가한 이유는 데이터베이스의 사용 때문이다. 쿠키값 생성의 시험에서는 데이터베이스를 사용하지 않고 임의의 데이터로 쿠키값을 생성하였다. 로그인 인증에서는 실제 사용자가 로그인을 위하여 입력한 아이디와 패스워드를 데이터베이스에서 확인하고, 확인된 사용자 정보로 쿠키값을 생성한다. 그리고 대한민국 개인정보보호법의 개인정보의 안전성 확보조치 기준 제8조에 따라 사용자의 IP주소, User Agent와 같은 접속기록을 데이터베이스에 저장한다[23]. Liu의 쿠키는 사용자의 접속기록을 데이터베이스에 저장할 때 IP주소와 User Agent만 저장하였지만, 강화된 안전쿠키에서는 쿠키 생성에 사용되는 쿠키값과 만료시간의 변화를 감지하기 위하여 해쉬값과 쿠키 만료시간도 함께 저장한다. Liu의 쿠키방식에서는 쿠키 생성을 위하여 HMAC-SHA256을 사용한 반면 강화된 안전쿠키에서는 SHA256을 사용하였다. 또한 데이터베이스에 한 쿼리문으로 접속기록뿐만 아니라 해쉬값과 쿠키 만료시간을 같이 저장하더라도 다른 쿠키방식과 시간차이가 거의 나지 않았다.

마지막으로 페이지 이동 시 쿠키검증에서는 Liu의 쿠키 방식에서는 각각 12.724ms의 시간이 걸렸고, 강화된 안전쿠키 방식에서는 약 21.435ms의 시간이 걸렸다. Liu의 쿠키와 다르게 강화된 안전쿠키에서 시간이 많이 소요된 이유는, 페이지 이동 시마다 쿠키를 생성 후 갱신된 쿠키와 만료시간을 다시 데이터베이스에 저장하는데 약 9ms의 시간이 추가되었다고 볼 수 있다. 갱신된 쿠키를 데이터베이스에 저장하는 것은 쿠키를 지속적으로 변하게 하여 재생공격을 차단하고, 짧은 시간이 추가되어 갱신된 만료시간은 재생공격을 차단할 뿐만 아니라 쿠키를 세션보다 안전하게 한다.

강화된 안전쿠키와 기존 쿠키의 성능을 비교하였을 때, 쿠키값의 생성과 로그인 인증에서는 거의 동일한 성능을 보였다. 사용자가 페이지 이동시의 쿠키검증에서는 새로 쿠키를 생성하는 과정에서 기존 쿠키에 비해 약 9ms의 시간이 증가하였다. M. C. Potter et al.[24]의 연구에 따르면, 인간의 뇌가 13ms 동안 보이는 이미지를 식별할 수 있음을 발견했고, 13ms 보다 적은 시간은 식별할 수 없었다고 한다. 따라서 9ms의 시간은 인간이 감지하기 힘든 짧은 시간이라 할 수 있다. 비록 기존 쿠키방식에 비하여 9ms의 시간이 증가하였지만, 강화된 안전쿠키 방식은 안전한 쿠키 체계의 모든 조건인 기밀성과 무결성, 인증 재생공격 방지를 모두 제공하고 있다.

V. 결 론

본 논문에서는 기존 연구[5][6]에서 만족하지 못하였던 안전한 쿠키체계의 조건인 기밀성과 무결성, 인증, 그리고 재생공격 방지를 모두 지원하는 강화된 안전쿠키를 제안하였다. ESCK는 사용자의 이름과 쿠키의 만료시간, 서버 비밀키를 연결하여 생성되는 해쉬값을 이용한다. 해쉬값을 구성하는 정보요소가 변조되었는지 확인이 가능하므로 무결성이 보장된다. 사용자의 데이터는 예측할 수 없고 길이가 긴 해쉬값을 키로 사용하여 대칭키 암호화되므로 ESCK에서는 기밀성이 보장된다. 사용자가 페이지를 이동할 때마다 쿠키를 검증하고 해쉬값과 쿠키 만료시간을 지속적으로 갱신하므로, 쿠키값이 노출되더라도 사용자에 의해 쿠키값과 쿠키의 만료시간이 변하게 되면 이전 쿠키값으로 사용자 인증을 할 수 없게 된다.

또한 쿠키의 생성 방법과 비밀키가 노출되더라도 쿠키값과 쿠키의 만료시간이 데이터베이스에 저장된 값과 같지 않으면 인증이 불가능하므로 보안적으로 안전하다. 쿠키의 만료시간을 짧게 설정하면 사용자가 쿠키값을 갱신하지 못하더라도 만료시간이 지난 쿠키는 인증할 수 없고, 공격자가 만료시간을 조작하더라도 쿠키의 만료시간이 데이터베이스에 저장되어 있으므로 인증할 수 없게 되어 재생공격으로부터 안전할 수 있다.

ESCK는 기존 연구[5][6]에 비해 쿠키값 생성시간과 로그인 인증에서는 비슷한 처리시간을 보였으며, 페이지 이동 시 쿠키검증 시험에서는 인간이 감지하기 힘든 시간인 약 9ms의 시간 증가에 불과한 성능을 보였다. 하지만 ESCK 방식은 기존 연구에 비해 안전한 쿠키 체계의 모든 조건인 기밀성과 무결성, 인증, 재생공격 방지를 모두 지원하고 있다. 만약 공격자가 쿠키의 만료시간 안에 쿠키를 탈취 및 재생하면 사용자 인증이 가능할 수 있다. 공격자가 만료시간 안에 쿠키를 탈취 및 재생하더라도 재생공격이 통하지 않는 쿠키 체계를 추후 연구할 계획이다.

References

[1] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee, "RFC2616: Hypertext Transfer Protocol — HTTP/1.1", RFC Editor, Jun. 1999. https://doi.org/10.17487/RFC2616.

[2] D. Kristol and L. Montulli, "RFC2109: HTTP State Management Mechanism", RFC Editor, Feb. 1997. https://doi.org/10.17487/RFC2109.

[3] O. Kulyk, N. Gerber, A. Hilt, and M. Volkamer, "Has the GDPR hype affected users' reaction to cookie disclaimers?", Journal of Cybersecurity, Vol. 6, No. 1, Dec. 2020. https://doi.org/10.1093/cybsec/tyaa022.

[4] J. S. Won, J. Park, and J.-G. Son, "A Defense Mechanism Based on Session Status against Cookie Replay Attack in Web Applications", KIPS Transactions on Computer and Communication Systems, Vol. 4, No. 1, pp. 31-36, Jan. 2015. https://doi.org/10.3745/KTCCS.2015.4.1.31.

[5] A. X. Liu, J. M. Kovacs, and M. G. Gouda, "A secure cookie scheme", Computer Networks, Vol. 56, No. 6, pp. 1723-1730, Apr. 2012. https://doi.org/10.1016/j.comnet.2012.01.013.

[6] W.-B. Lee, H.-B. Chen, S.-S. Chang, and T.-H. Chen, "Secure and efficient protection for HTTP cookies with self-verification", International Journal of Communication Systems, Vol. 32, No. 2, Jan. 2019. https://doi.org/10.1002/dac.3857.

[7] A. Cahn, S. Alfeld, P. Barford, and S. Muthukrishnan, "An Empirical Study of Web Cookies", In Proc. of the 25th International Conference on World Wide Web (WWW '16), Montréal Québec Canada, pp. 891-901, Apr. 2016. https://doi.org/10.1145/2872427.2882991.

[8] PHP.net, https://www.php.net/manual/en/ini.core.php [accessed: Jan. 08, 2024].

[9] Tutorials Point, https://www.tutorialspoint.com/What-is-the-maximum-size-of-a-web-browser-s-cookies-value [accessed: Jan. 08, 2024].

[10] K. Fu, E. Sit, K. Smith, and N. Feamster, "Dos and don'ts of client authentication on the web", In Proc. of the 10th conference on USENIX Security Symposium, Vol. 10, pp. 19, Aug. 2001. https://doi.org/abs/10.5555/1267612.1267631.

[11] J. S. Park and R. S. Sandhu, "Secure cookies on the web", IEEE Internet Computing, Vol. 4, No. 4, pp. 36-44, Jul. 2000. https://doi.org/10.1109/4236.865085.

[12] A. Barth, "HTTP State Management Mechanism", RFC 6265, Apr. 2011. https://doi.org/10.17487/RFC6265.

[13] S. Barbato, S. Dorigotti, and T. Fossati, "SCS: KoanLogic's Secure Cookie Sessions for HTTP", RFC 6896, Mar. 2013. https://doi.org/10.17487/RFC6896.

[14] OWASP, "A7:2017-Cross-Site Scripting (XSS)", https://owasp.org/www-project-top-ten/2017/A7_2017-Cross-Site_Scripting_(XSS) [accessed: Jan. 08, 2024].

[15] W3Techs, "Usage statistics and market share of WordPress", https://w3techs.com/technologies/details/cm-wordpress [accessed: Jan. 08, 2024].

[16] Wordpress, https://developer.wordpress.org/reference/functions/wp_generate_auth_cookie/ [accessed: Jan. 08, 2024].

[17] W3Techs, "Usage statistics of Default protocol https for websites", https://w3techs.com/technologies/details/ce-httpsdefault [accessed: Jan. 08, 2024].

[18] R. C. Merkle, "Secrecy, Authentication and Public Key Systems", Ph.D. thesis, Department of Electrical Engineering, Stanford University, pp. 187, 1979. https://dl.acm.org/doi/10.5555/909000.

[19] B. Preneel, "Collision resistance", Encyclopedia of Cryptography and Security, pp. 81-82, 2005. https://doi.org/10.1007/0-387-23483-7_69.

[20] C. Liu, R. W. White, and S. Dumais, "Understanding web browsing behaviors through Weibull analysis of dwell time", In Proc. of the 33rd international ACM SIGIR conference on Research and development in information retrieval, pp. 379-386, Jul. 2010. https://doi.org/10.1145/1835449.1835513.

[21] GDPR, "Cookies, the GDPR, and the ePrivacy Directive", https://gdpr.eu/cookies/ [accessed: Jan. 08, 2024].

[22] PHP,net, "Runtime Configuration", https://www.php.net/manual/en/session.configuration.php [accessed: Jan. 08, 2024].

[23] The Ministry of Public Administration and Security (Personal Information Protection Policy), "Basic Measures for Securing Safety of Personal Information", 2020.

[24] M. C. Potter, B. Wyble, C. E. Hagmann, and E. S. McCourt, "Detecting meaning in RSVP at 13 ms per picture", Atten Percept Psychophys, Vol. 76, pp. 270-279, 2014. https://doi.org/10.3758/s13414-013-0605-z.

저자소개

정 진 호 (Jin-Ho Jeong)

2019년 : 안동대학교 컴퓨터공학과(공학사)
2021년 : 안동대학교 컴퓨터공학과(공학석사)
2021년 ~ 현재 : 안동대학교 컴퓨터공학과 박사과정
2015년 ~ 현재 : 디제이패밀리 대표
관심분야 : 웹보안

차 영 욱 (Young-Wook Cha)

1987년 : 경북대학교 전자공학과 (공학사)
1992년 : 충남대학교 전자통계학과 (공학석사)
1998년 : 경북대학교 컴퓨터공학과 (공학박사)
1987년 ~ 1999년 : 한국전자통신 연구원 선임연구원
2003년 ~ 2004년 : 매사추세츠 주립대학 방문학자
2019년 ~ 현재 : 안동대 사이보보안 센터장
1999년 ~ 현재 : 안동대학교 컴퓨터공학과 교수
관심분야 : 망/시스템 제어 및 관리, 블록체인 및 보안, 개방형통신망