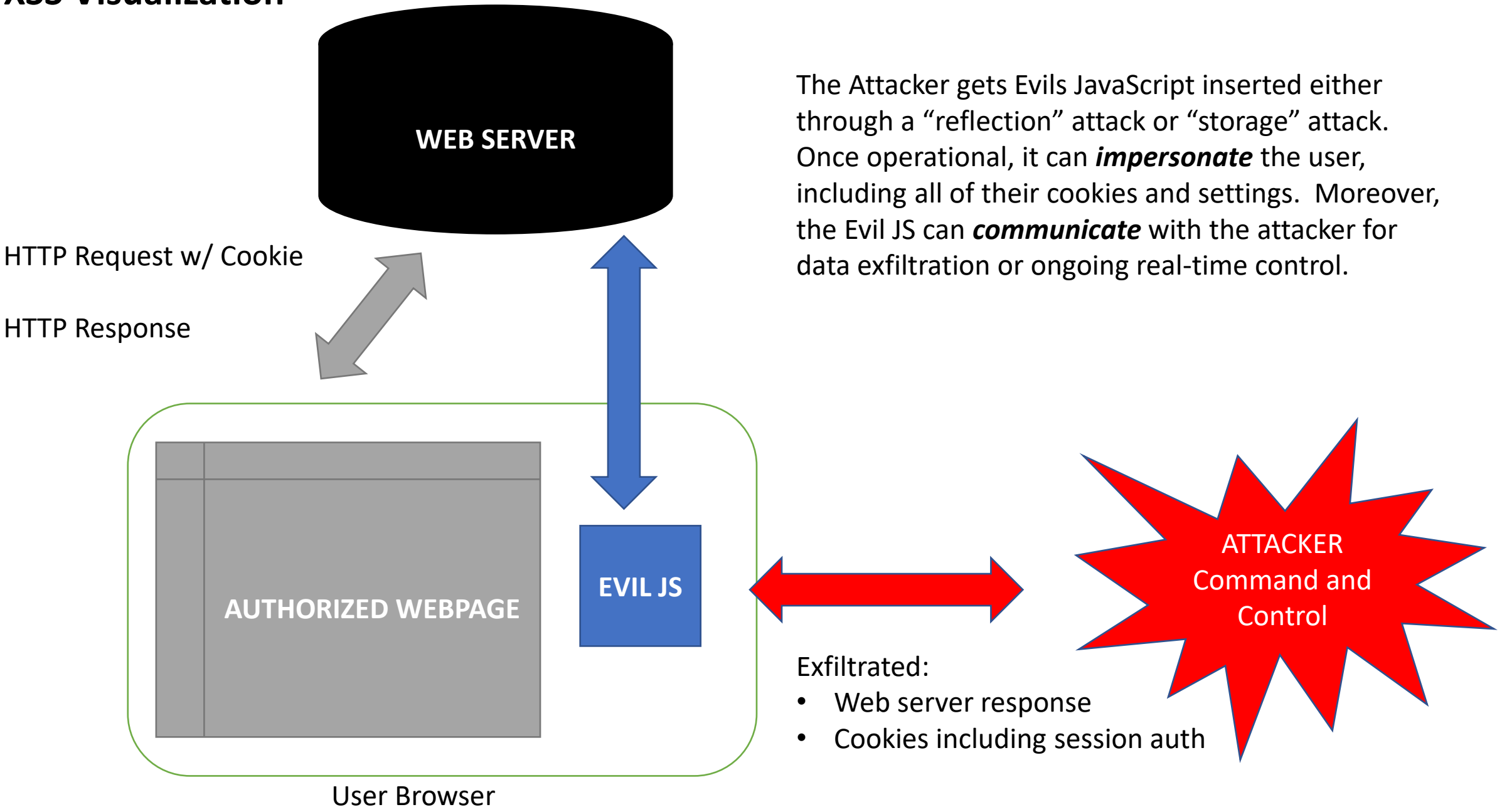
XSS, CSRF Supplement

XSS Visualization

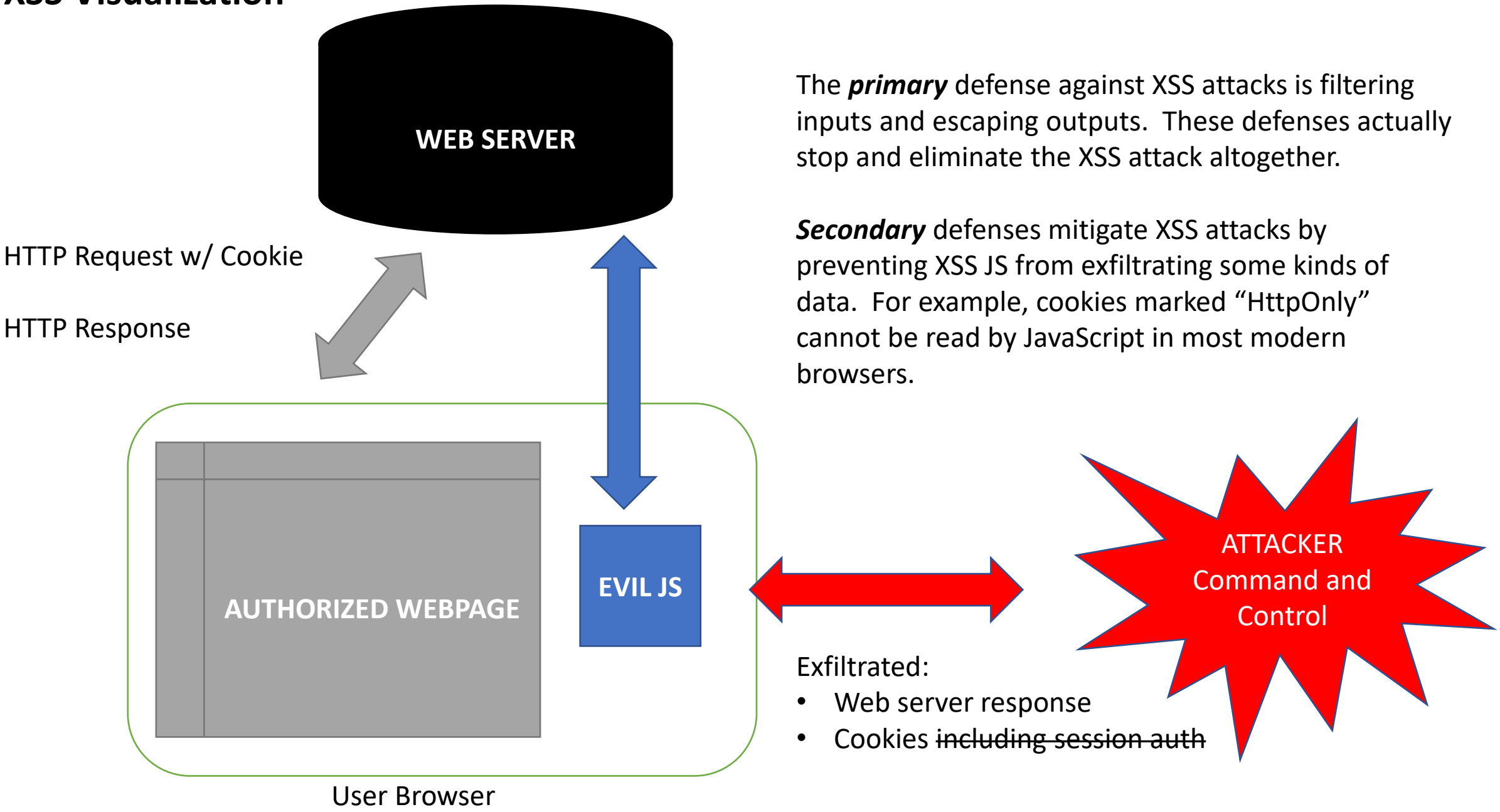


The Attacker gets Evils JavaScript inserted either through a “reflection” attack or “storage” attack. Once operational, it can ***impersonate*** the user, including all of their cookies and settings. Moreover, the Evil JS can ***communicate*** with the attacker for data exfiltration or ongoing real-time control.

Exfiltrated:

- Web server response
- Cookies including session auth

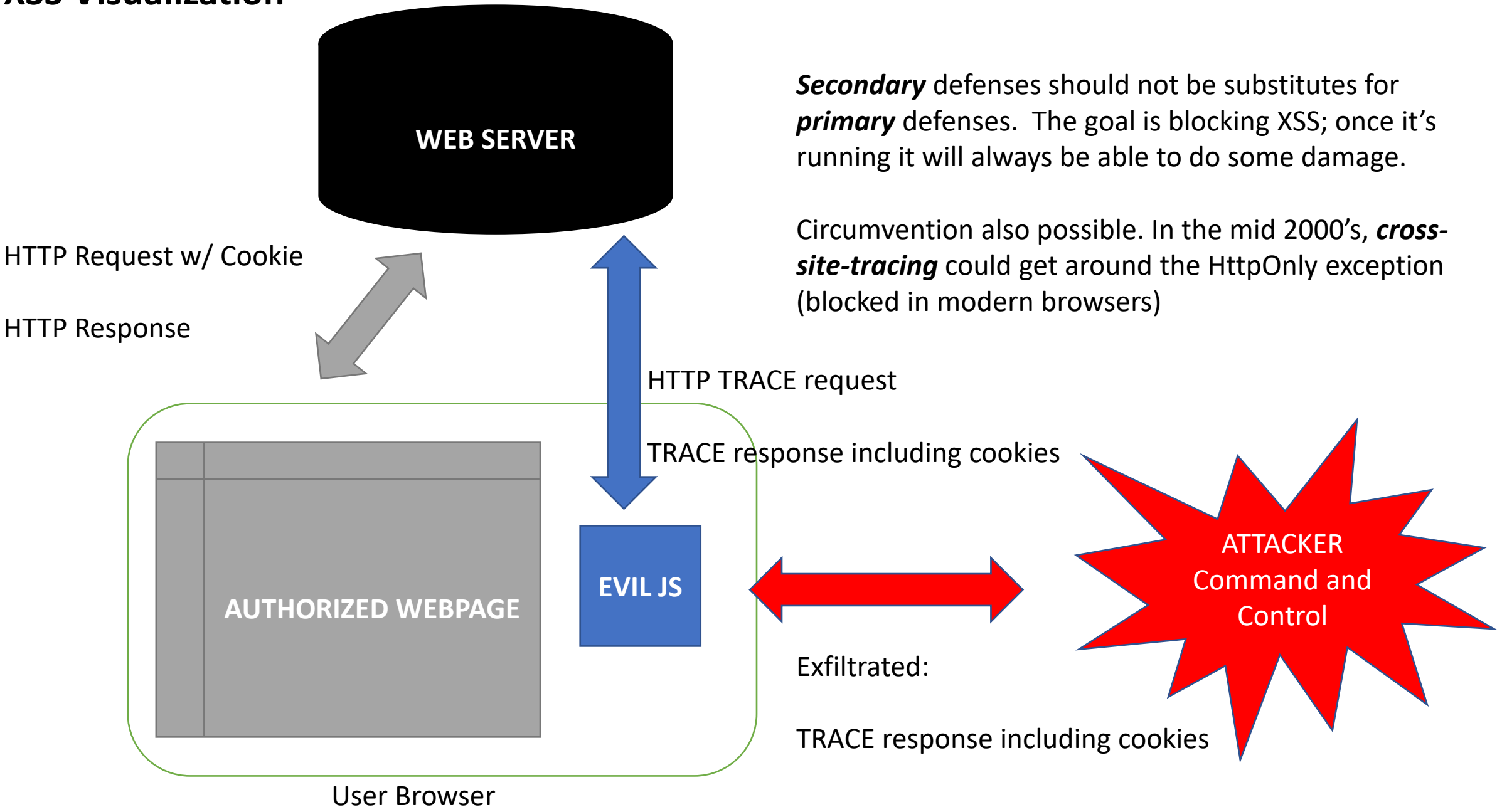
XSS Visualization



The ***primary*** defense against XSS attacks is filtering inputs and escaping outputs. These defenses actually stop and eliminate the XSS attack altogether.

Secondary defenses mitigate XSS attacks by preventing XSS JS from exfiltrating some kinds of data. For example, cookies marked "HttpOnly" cannot be read by JavaScript in most modern browsers.

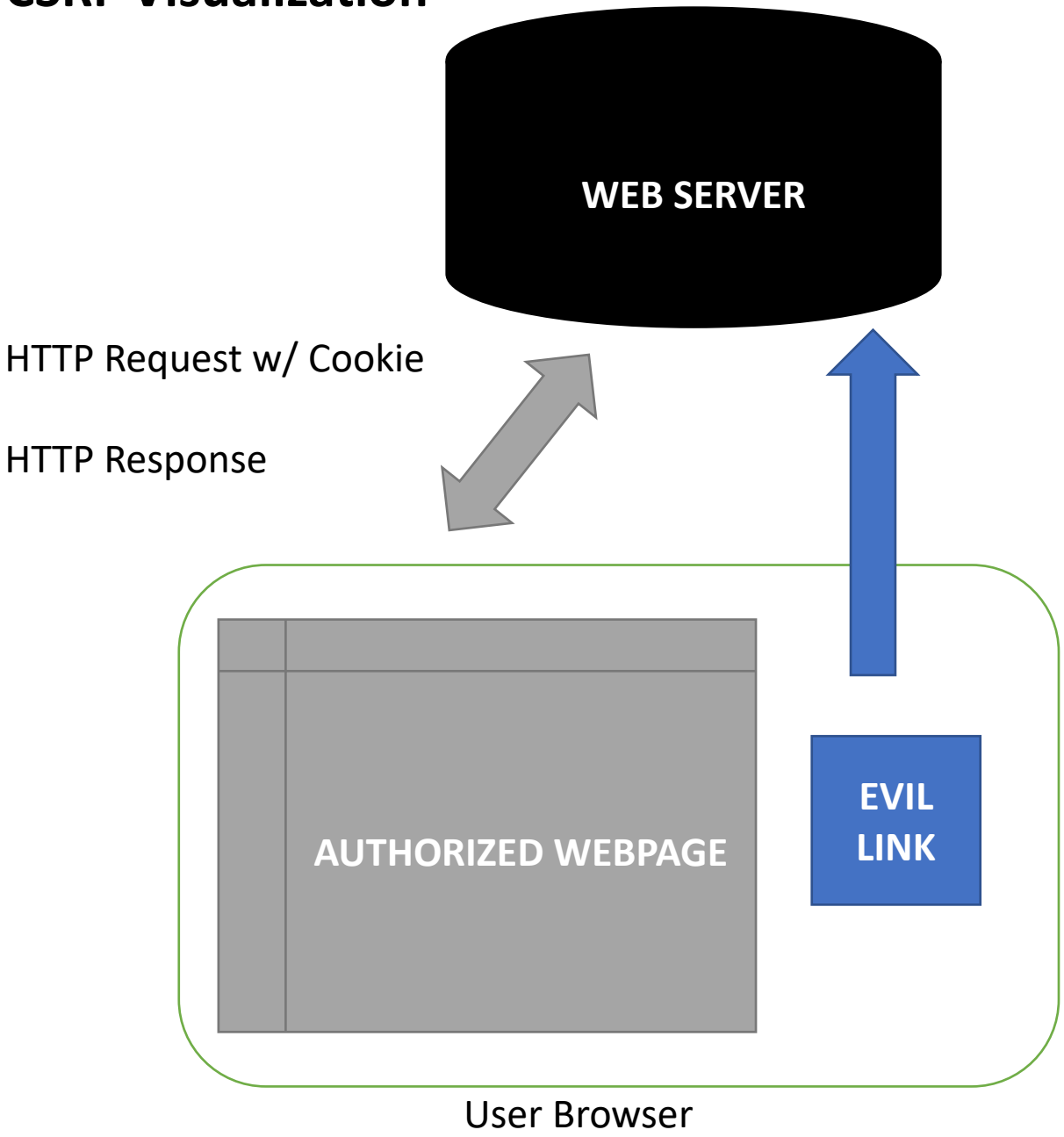
XSS Visualization



Secondary defenses should not be substitutes for ***primary*** defenses. The goal is blocking XSS; once it's running it will always be able to do some damage.

Circumvention also possible. In the mid 2000's, ***cross-site-tracing*** could get around the HttpOnly exception (blocked in modern browsers)

CSRF Visualization

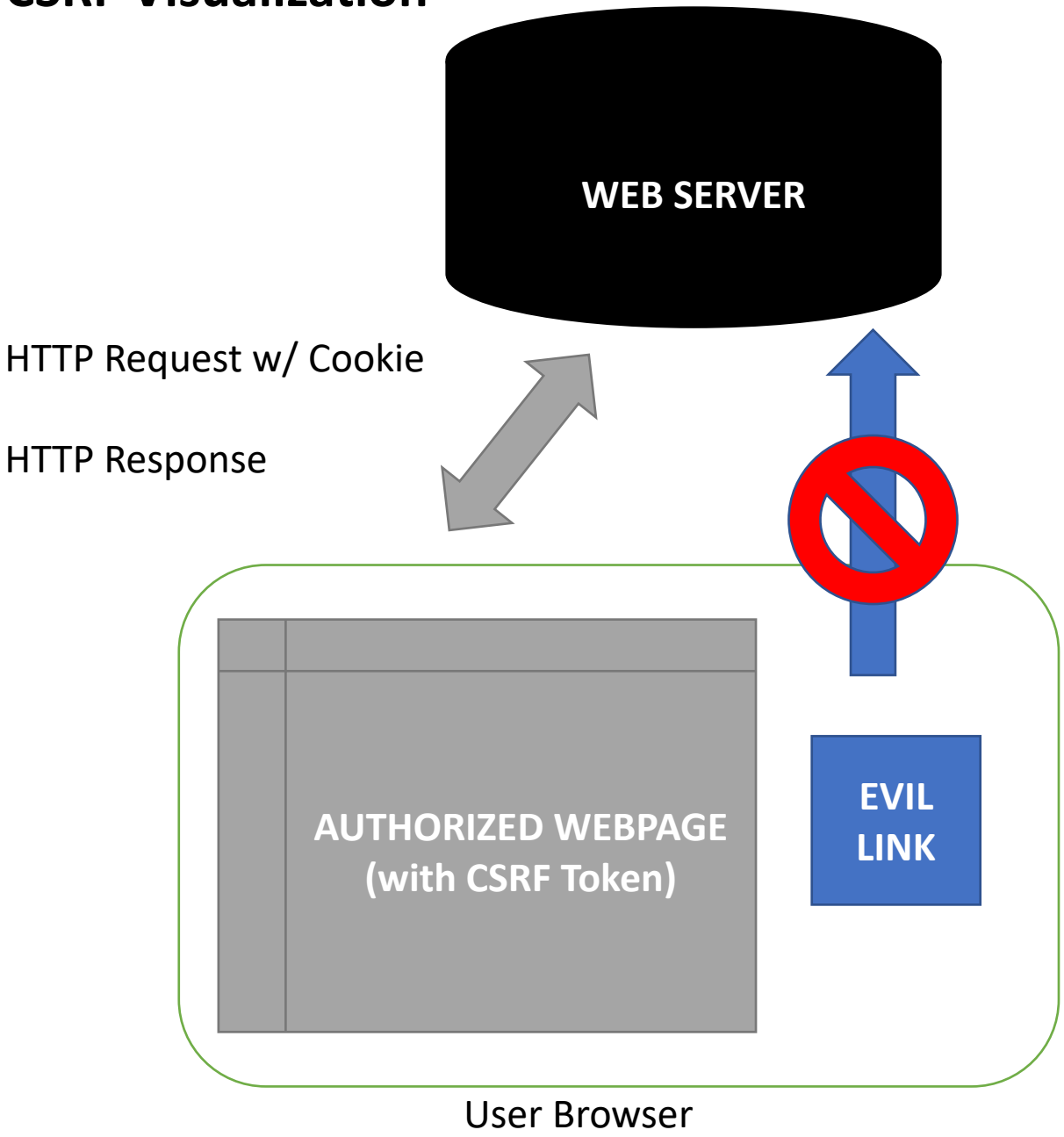


Cross-Site Request Forgery is simpler than XSS. There is typically no JS and it is not typically ***two-way communication with the Attacker***.

The idea is simply getting the victim to click on a link or otherwise transmit an HTTP request that causes an unauthorized transaction. For the attacker to succeed:

1. An inducible action
2. Cookie-based session handling
3. Predictable request parameters

CSRF Visualization

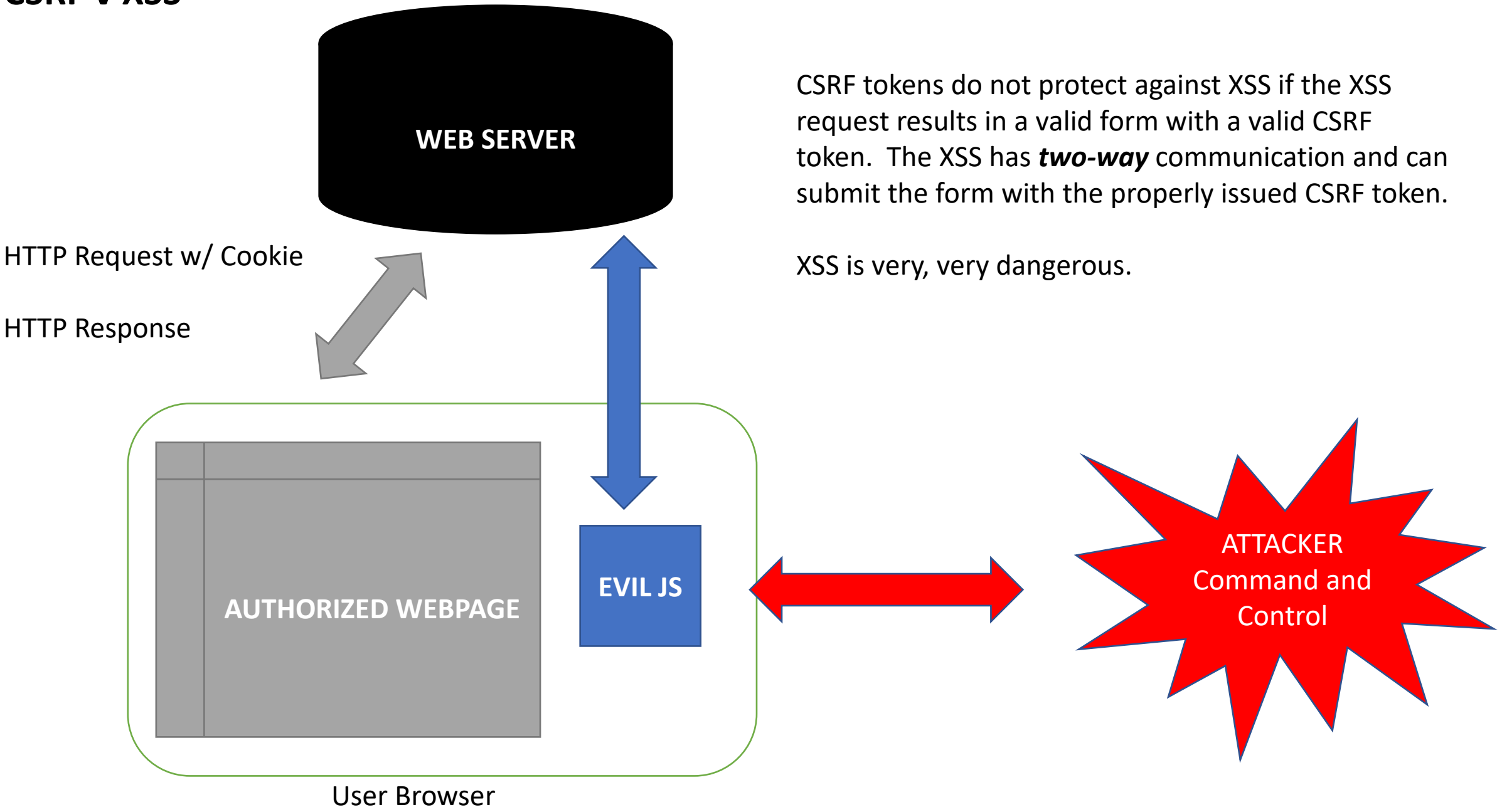


A ***CSRF-Token*** is some ***unpredictable*** value embedded in the webpage that is used for identifying authorized requests. For this to work:

1. CSRF Token cannot be a cookie
2. Must be unpredictable
3. Not easily interceptable

Typically issued from the server in a hidden form element. Automatically transmitted back when the form is submitted.

CSRF v XSS



CSRF tokens do not protect against XSS if the XSS request results in a valid form with a valid CSRF token. The XSS has ***two-way*** communication and can submit the form with the properly issued CSRF token.

XSS is very, very dangerous.