

Audyty bezpieczeństwa aplikacji Internetowej na potrzeby realizacji pilotażowego programu „Profilaktyki krótkowzroczności u dzieci z klas 1-3”

Procedura i narzędzia wykorzystane do realizacji audytu

Oprogramowaniem, którym posłużono się do realizacji audytu jest narzędzie „OWASP ZAP” w wersji 2.11.0.

OWASP ZAP (Zed Attack Proxy) jest projektowany na licencji typu open-source przez organizację OWASP (Open Web Application Security Project, zajmującą się bezpieczeństwem w sieci Internet).

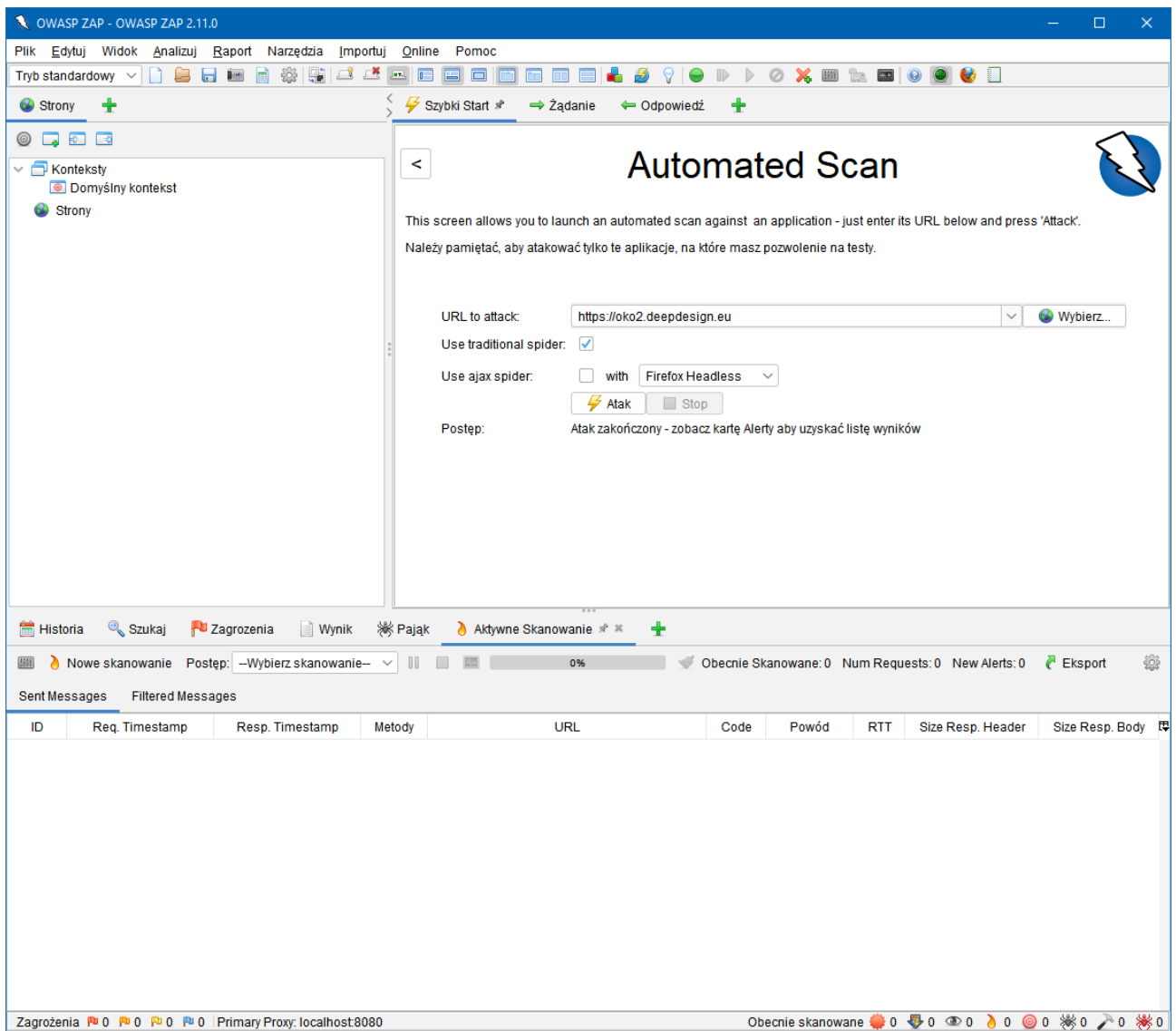
Adres projektu: <https://www.zaproxy.org>

Adres projektu w OWASP: <https://owasp.org/www-project-zap>

Interfejs programu jest bardzo intuicyjny. W polu „url to attack” wpisujemy adres internetowy jaki chcemy „zaatakować” (czyli jaki chcemy tak naprawdę zbadać pod kątem podatności na różnego rodzaju ataki). Po wprowadzeniu ścieżki url naciskamy przycisk „Attack”. Do „ataku” na aplikację internetową został wykorzystany tradycyjny pająk sprawdzający między innymi podatność na:

- **Cross Site Scripting (XSS).**
- **SQL injection.**
- **Wykonanie kodu ze „złośliwego pliku” (Malicious File Execution).**
- **Bezpośrednie odwołania w kodzie.**
- **Obchodzenie ścieżki.**
- **Zabezpieczenia Cookies.**
- **Dostęp do prywatnych zasobów.**

Oczywiście badanych podatności jest o wiele więcej, o których można przeczytać na stronie internetowej zawierającą dokumentację projektu.



Rys. 1. Interfejs OWASP ZAP.

Audyt

The screenshot shows the OWASP ZAP 2.11.0 interface. The main window is titled 'Automated Scan' and contains the following text: 'This screen allows you to launch an automated scan against an application - just enter its URL below and press 'Attack''. Below this, there is a form with the following fields and options:

- URL to attack: - Use traditional spider:
- Use ajax spider: with
- Buttons:
- Postęp: Atak zakończony - zobacz kartę Alerty aby uzyskać listę wyników

At the bottom of the interface, there is a table with the following columns: ID, Req. Timestamp, Resp. Timestamp, Metody, URL, Code, Powód, RTT, Size Resp. Header, and Size Resp. Body. The table contains 17 rows of scan results.

ID	Req. Timestamp	Resp. Timestamp	Metody	URL	Code	Powód	RTT	Size Resp. Header	Size Resp. Body
1819	23.11.2021, 11:49:41	23.11.2021, 11:49:41	GET	https://oko2.deepdesign.eu/system/default/con...	301	Moved Perm...	30 ms	294 bajtów	273 bajtów
1820	23.11.2021, 11:49:41	23.11.2021, 11:49:41	GET	https://oko2.deepdesign.eu/system/default/css	301	Moved Perm...	30 ms	275 bajtów	254 bajtów
1821	23.11.2021, 11:49:41	23.11.2021, 11:49:41	GET	https://oko2.deepdesign.eu/system/default/js	301	Moved Perm...	30 ms	274 bajtów	253 bajtów
1822	23.11.2021, 11:49:41	23.11.2021, 11:49:41	GET	https://oko2.deepdesign.eu/app	301	Moved Perm...	30 ms	268 bajtów	247 bajtów
1823	23.11.2021, 11:49:41	23.11.2021, 11:49:42	POST	https://oko2.deepdesign.eu/autoryzacja	200	OK	54 ms	463 bajtów	7 718 bajtów
1824	23.11.2021, 11:49:41	23.11.2021, 11:49:42	GET	https://oko2.deepdesign.eu/app/composer	301	Moved Perm...	31 ms	295 bajtów	278 bajtów
1825	23.11.2021, 11:49:42	23.11.2021, 11:49:42	GET	https://oko2.deepdesign.eu/app/composer/ven...	301	Moved Perm...	29 ms	276 bajtów	255 bajtów
1826	23.11.2021, 11:49:42	23.11.2021, 11:49:42	POST	https://oko2.deepdesign.eu/autoryzacja	200	OK	39 ms	463 bajtów	7 718 bajtów
1827	23.11.2021, 11:49:42	23.11.2021, 11:49:42	GET	https://oko2.deepdesign.eu/app/composer/ven...	301	Moved Perm...	27 ms	287 bajtów	266 bajtów
1828	23.11.2021, 11:49:42	23.11.2021, 11:49:42	POST	https://oko2.deepdesign.eu/autoryzacja	200	OK	48 ms	463 bajtów	7 718 bajtów
1829	23.11.2021, 11:49:42	23.11.2021, 11:49:42	GET	https://oko2.deepdesign.eu/app/composer/ven...	301	Moved Perm...	26 ms	294 bajtów	273 bajtów
1830	23.11.2021, 11:49:42	23.11.2021, 11:49:42	POST	https://oko2.deepdesign.eu/autoryzacja	200	OK	46 ms	463 bajtów	7 718 bajtów
1831	23.11.2021, 11:49:42	23.11.2021, 11:49:42	POST	https://oko2.deepdesign.eu/autoryzacja	200	OK	40 ms	463 bajtów	7 718 bajtów
1832	23.11.2021, 11:49:42	23.11.2021, 11:49:42	POST	https://oko2.deepdesign.eu/autoryzacja	200	OK	50 ms	463 bajtów	7 718 bajtów
1833	23.11.2021, 11:49:42	23.11.2021, 11:49:42	POST	https://oko2.deepdesign.eu/autoryzacja	200	OK	49 ms	463 bajtów	7 718 bajtów

Rys. 2. Okno po skończonym audycie (100%).

Lista ścieżek które atakował pajak dostępna jest w załączniku niniejszego audytu („adresy-pajak.csv”).

Audyty wykazał:

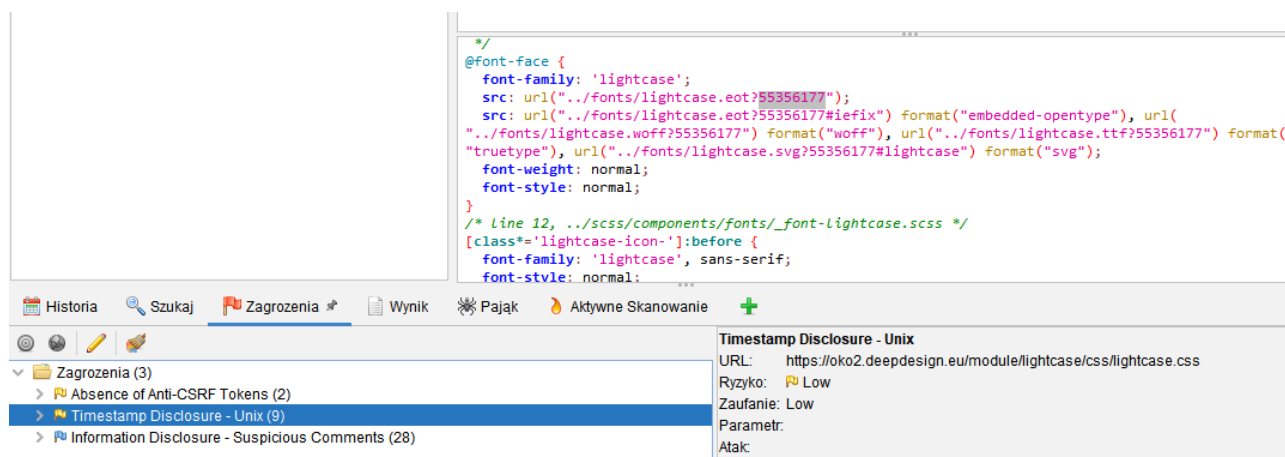
- brak podatności wysokiego ryzyka
- brak podatności średniego ryzyka,
- dwie podatności niskiego ryzyka.

Interpretacja typów podatności niskiego ryzyka

Timestamp Disclosure

Podatność Timestamp Disclosure dotyczy zagrożenia w momencie wycieku np. znaczników czasu po stronie serwera obsługujących wrażliwe dane. Na poniższym zrzucie widać, że diagnoza narzędzia dotyczy pliku CSS (style layoutu po stronie klienta). Nie są to dane absolutnie dające możliwość zaatakowania aplikacji. Tak więc zgodnie z zapisem na stronie <https://scanrepeat.com/web-security-knowledge-base/timestamp-disclosure---unix>

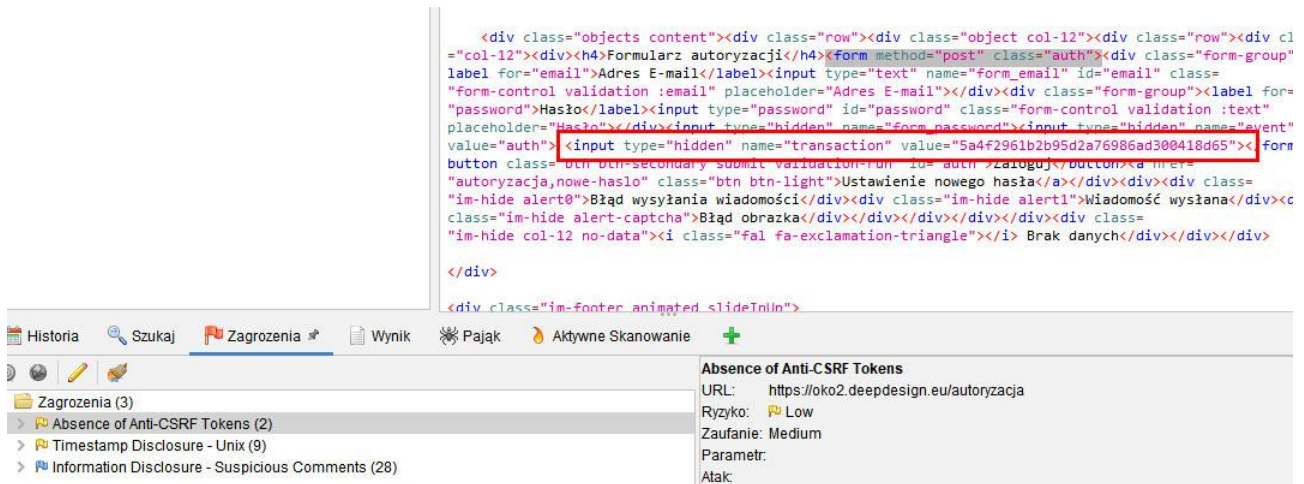
Podatność w tym przypadku można zignorować: „*If a given Timestamp Disclosure alert is not critical it can be ignored*”.



Rys. 3. Timestamp Disclosure.

Absence of Anti-CSRF Tokens

Każdy formularz (nie tylko autoryzacji) w aplikacji jest zabezpieczony losowym tokenem, który po stronie serwera jest weryfikowany. Zrzut poniżej przedstawia podgląd kodu formularza z zaznaczoną ukrytą zmienną „tokenową”.



Rys. 4. Absence of Anti-CSRF Tokens.

Taki zabieg zapewnia autentyczność każdego przesyłanego w aplikacji formularza.

Na poniższym zrzucie widać jak po stronie serwera zmienna „transaction” jest weryfikowana i dodawana do tablicy już zrealizowanych (dwa identyczne przesłania formularza nie mogą się odbyć, w razie odświeżenia strony lub powrotu). Jeżeli nastąpi brak zmiennej „tokenowej”, to formularz się „nie zrealizuje”.

Plik: `php/run/init.php`

```
//First operation
if($session->getSession('transaction') === '')
    $session->setSession('transaction', array());

//Operations
if(!in_array($p_transaction, $session->getSession('transaction'))) {...}
```

...

```
//No repeat
if($p_transaction and !in_array($p_transaction, $session-
>getSession('transaction')))
    $session->pushSession('transaction', $p_transaction);
```

Wracając do wskazania podatności przez narzędzie audytujące. Wskazanie to może zostać zignorowane z uwagi na powyższy dowód. Na pewno dodanie mechanizmu captcha (tak jak w formularzu resetu hasła, który nie wskazuje podatności) wyeliminuje zastrzeżenie,

aczkolwiek, proces autoryzacji będzie wówczas bardziej złożony (wpisywanie dodatkowych losowych znaków).

Należy pamiętać, że te dwa wskazania audytu są o **niskim zagrożeniu**, które zostały jasno podważone powyżej.

Podsumowanie

Niniejszy audyt wykazał, że aplikacja internetowa jest zabezpieczona przed potencjalnymi atakami. Należy również pamiętać o dodatkowych zabezpieczeniach jakie zostały zaimplementowane po stronie serwera (`password_hash()`, `password_verify()`) i samej pseudonimizacji danych. Więcej o tych zagadnieniach w dokumentacji aplikacji internetowej.

Opracowanie: Damian Krawiec
Zielona Góra, 23.11.2021