Wymagania bezpieczeństwa dla Systemu eAtesty

SPIS TREŚCI

I.WYMAGANIA BEZPIECZEŃSTWA DLA APLIKACJI INTERNETOWEJ

I.1 ZASADY GENERALNE

Aplikacja internetowa powinna zapewniać odporność co najmniej na wskazane podatności w wykazie **Top 10 Web Application Security Risc w wersji 2021** organizacji **OWASP ®FOUNDATION .**

Jeśli w trakcie realizacji projektu zostanie ujawniona nowa istotna podatność to Wykonawca powinien, jeśli to będzie możliwe, zapewnić jej usunięcie lub moderowanie środkami technicznymi bądź zalecić odpowiednie środki organizacyjne.

Wykonawca powinien dowieść powyższego zarówno na etapie **PROJEKTU TECHNICZNEGO** w dedykowanym rozdziale **ZAŁOŻENIA BEZPIECZEŃSTWA APLIKACJI** który podlega ocenie Zamawiającego jak i poprzez wykonanie testów penetracyjnych zarówno zautomatyzowanych jak i manualnych których wyniki przedstawi w postaci pisemnego raportu zawierającego co najmniej:

1. Ogólny opis metodyki testów
2. Szczegółowy opis metodyki testów z wskazaniem sposobu weryfikacji każdej możliwej podatności o której mowa w **OWASP TOP 10**
3. Skład personalny zespołu testerów z wskazaniem wymaganych certyfikatów
4. Wykaz narzędzi użytych do przeprowadzenia testów
5. Wyniki testów w postaci zrzutu ekranów i części opisowej
6. Klasyfikacja zidentyfikowanych podatności wg wskaźnika CVSS (Common Vulnerability Scoring System w wersji co najmniej 3.1)
7. Ocena końcowa uwzględniająca zakres ocen CVSS w sposób następujący:
    a. Krytyczny – niedopuszczający do stosowania
    b. Wysoki -niedopuszczajacy do stosowania
    c. Średni- warunkowo dopuszczony, wymagany termin usunięcia lub moderacja
    d. Niski- dopuszczony, zalecane określenie terminu usunięcia lub moderacja

I.2 ZASADY SZCZEGÓŁOWE

Przyjmując nadrzędność zasad wskazanych w rozdziale **ZASADY GENERALNE** w szczególności rozważy poniższe zagrożenia i opisane metody zapobiegania. Niektóre z wskazanych metod obrony są właściwe dla konkretnych technologii. Wykonawca jest zobowiązany wybrać właściwe dla przyjętej przez siebie technologii. W tych frameworkach które automatycznie wykorzystują opisane poniżej metody obrony należy zweryfikować czy rzeczywiście są one wykorzystywane. Pozytywne spełnienie poniższych nie zwalnia Wykonawcy z kompletnego wykonania zadań określonych w **ZASADACH GENERALNYCH**.

1. **Wstrzykiwanie SQL-a (org. SQL injection)**

   a. Obrona przez użycie zmiennych wiązanych (instrukcje parametryzowane)
   b. Obrona przez mapowanie obiektowo-relacyjne org. (object relating mapping)
   c. Obrona przez minimalizację uprawnień DML
2. **Wstrzykiwanie kodu poleceń (code injection)**
   a. Obrona przez sekwencję ucieczki dla znaków sterujących (escape control char)
   b. Obrona przez konwersję typu (łańcuch na tablicę)
3. **Zdalne wykonanie kodu (org. Remote Code Execution)**
   a. Obrona przez weryfikację reputacji bibliotek do deserializacji w autorytatywnych źródłach
4. **Przesłanie złośliwego pliku**
   a. Blokada możliwości wykonania zawartości pliku
   b. Weryfikacja kontentu
   c. Bezpieczny hosting zewnętrzny (sieci klasy CDN)
5. **Zapisany atak XSS (org.Stored Cross-Side Scripting**)
   a. Obrona przez użycie encji nazwanych dla znaczników HTML (sekwencja ucieczki od znaków kontrolnych HTML)
   b. Obrona przez wdrożenie zasad bezpieczeństwa treści (org. Establishing Content-Security Policy)
6. **Odbity atak XSS (org. Reflected Cross-Side Scripting)**
   a. Obrona przez użycie encji nazwanych dla znaczników HTML w treści przychodzącej od użytkowników (sekwencja ucieczki od znaków kontrolnych HTML)
7. **Cross-Side Scripting oparty na hierarchii DOM (interpretacja przez Java Script przyrostka URI na przeglądarce)**
   a. Stosowanie sekwencji ucieczki
   b. Wykorzystanie dojrzałych platform zabezpieczonych natywnie przed tym mechanizmem
8. **Atak przesiadkowy (org.Cross-Side Request Forgery- CSRF)**
   a. Obrona przez stosowanie zasad REST (Representional State Transfer- brak możliwości zmiany stanu serwera przez wykorzystanie żądania GET)
   b. Użycie tokenów CSRF wraz z zasadą SameSite (Strict lub Lax dla zachowania płynności tam gdzie jest to niezbędne)
9. **Atak na uwierzytelnienie (org. Authentication)**
   a. Unikanie metod natywnych http ,przede wszystkim metody *base*
   b. Wykorzystanie uznanego przez autorytatywne źródła usług zewnętrznych (OAuth, OPEN ID etc.)

Projekt pn. „E-ATESTY uruchomienie e-usług za pośrednictwem dedykowanej aplikacji mobilnej wspieranej interoperacyjną platformą informatyczną” współfinansowanego ze środków EFRR w ramach POPC nr POPC.02.04.00-00-0089/20-00

c. Wykorzystanie środków identyfikacji elektronicznej certyfikowanych przez kraje unijne na poziomie co najmniej średnim (zgodnie z rozporządzeniem wykonawczym wydanym do eIDAS), dostępnych poprzez Węzeł Krajowy
d. Weryfikacja syntaktyki i autentyczności posiadania adresów e-mail użytkownika przy implementacji własnych mechanizmów logowania
e. Identyfikacja i odrzucanie tzw. adresów tymczasowych (org. temporary adres)
f. Bezpieczne metody resetowania hasła (krótki termin ważności oraz natychmiastowe unieważnienie po użyciu)
g. Bezpieczne przechowywanie haseł na serwerach – wyłącznie hash z solą
h. Wykorzystanie HSM do generowania sekretów (Hardware Security Module - **opcja** , bardzo kosztowne rozwiązanie)
i. Wprowadzenie uwierzytelnienia wieloskładnikowego (kilka wariantów do wyboru użytkownika – n.p. Google Authenticator, MS Authenticator etc.)
j. Implementowanie mechanizmu „testu Turinga" – CAPTCHA (obecnie uznany za niepewny wskutek rozwoju AI)
k. Zapewnienie prostego mechanizmu wylogowania się z sesji

10. Atak na otwartą sesję

a. Cookie sesji zabezpieczone w odpowiedzi http poprzez słowa kluczowe:
    i. **HttpOnly**
    ii. **Secure**
    iii. **SameSite=Strict** lub dla zachowania płynności w dostępie przez linki **SameSite=Lax**
b. Wyłączenie mechanizmu przepisywania URL (zapobieżenie tzw. fiksacji sesji)
c. Wymuszenie protokołu HTTPS (obrona przed MITM)

11. Atak na uprawnienia (podnoszenie uprawnień – pionowe i poziome)

a. Wybór uznanego modelu reguł autoryzacji
    i. Listy kontroli dostępu (org. Acccess control lists – ACL)
    ii. Kontrola dostępu oparta na rolach (Role-Based Acccess Control – RBAC)
    iii. Białe i czarne listy(Black and White Lists) – metoda prosta, lecz niekompletna (wiele kont może być w stanie nieokreślonym)
b. Wielopoziomowa kontrola uwierzytelnienia (zasada Zero-Trust)
c. Kodowanie identyfikatorów dostępu do wrażliwych zasobów (wymaga bazy identyfikator – ścieżka do zasobu) – zapobieganie atakom directory traversal
d. Użycie wyrażeń regularnych do odfiltrowania łańcuchów o składni ścieżki – zapobieganie jak wyżej

12. Ataki na obiekty szyfrowane

a. Używanie bezpiecznych protokołów ochrony kanału transmisji – aktualnie TLS co najmniej ver. 1.2

b. Zaufanie tylko do wiarygodnych wydawców certyfikatów- zapewnienie autentyczności domeny
c. Certyfikaty z podpisem własnego PKI tylko dla systemów wewnętrznych (testowych)
d. Wymuszenie protokołu HTTPS poprzez ustawienie nagłówka Strict-Transport-Security w odpowiedzi serwera
e. Używanie tylko uznanych za bezpieczne algorytmów szyfrowania do ochrony obiektów w bazach

13. Ataki na podatne biblioteki

a. Kontrola wersji użytych bibliotek poprzez wykorzystanie narzędzi do budowy drzewa zależności pomiędzy użytymi bibliotekami (z oznaczonymi wersjami)
b. Weryfikacja informacji o podatnych wersjach bibliotek
c. Zapewnienie integralności uzytych bibliotek poprzez sumy kontrolne i ich weryfikację (implementacja w przeglądarce na podstawie konstrukcji HTML-a)
d. Instalowanie poprawek lub aktualnych (niepodatnych) wersji tak szybko jak to jest możliwe
e. Regularne wykonywanie audytów i testów penetracyjnych ukierunkowanych na powyższą podatność

14. Ataki na słabą (niewłaściwą) konfigurację

a. Zmiana domyślnych danych dostępowych (również w środowisku testowym)
b. Blokowanie możliwości indeksowania katalogów
c. Ochrona wrażliwych parametrów konfiguracyjnych serwera poprzez umieszczanie w szyfrowanych kontenerach własnych lub autorytatywnych serwisach zewnętrznych
d. Wyłączanie zbędnych serwisów administracyjnych
e. *Hardening* konfiguracji środowisk produkcyjnych i odpowiednio testowych
f. Zmiana kluczy szyfrujących i kluczy API po przejściu do środowiska produkcyjnego

15. Ataki na XML (Denial of Services, XML External Entity)

a. Wykluczenie przetwarzania plików DTD (Document Type Definition)
b. Używanie bibliotek uniemożliwiających przetwarzanie XML właściwych dla danych języków (Java, .NET, Node.js, Python etc.)

II. WYMAGANE KWALIFIKACJE PERSONELU WYKONAWCY

II.1 Kwalifikacje Zespołu Projektowego

Wymaga się aby założenia **Architektury Bezpieczeństwa** były zatwierdzone przez osobę posiadającą ważny certyfikat CISSP (*Certified Information Systems Security Profesional*) lub CISM (*Certified Information Security Manager*) lub równoważny.

II.2 KWALIFIKACJE ZESPOLU TESTOWEGO

Wymaga się aby testy penetracyjne o zakresie uzgodnionym z Zamawiającym zostały przeprowadzone przez Zespół w którym co najmniej dwie osoby posiadają certyfikat OSCP (*Offensive Security Certified Proffesional*) lub równoważny. Wyniki przeprowadzonego testu podpisują obie wymienione osoby.