PTER - metodyka testowania bezpieczeństwa aplikacji internetowych

Marek ANTCZAK, Zbigniew ŚWIERCZYŃSKI

Instytut Teleinformatyki i Automatyki WAT,
ul. Kaliskiego 2, 00-908 Warszawa
marek.antczak@gmail.com, z.swierczynski@ita.wat.edu.pl

STRESZCZENIE: Artykuł opisuje główne założenia opracowanej metodyki testów bezpieczeństwa aplikacji internetowych (WWW). Metodyka koncentruje się na opisie sposobu przeprowadzenia testów tzw. „ścieżki technicznej" oraz dokumentowania ich wyników. Dzięki zastosowanemu "zwinnemu" podejściu metodyka pozwala skoncentrować się na rzeczywistych potrzebach odbiorcy testów bezpieczeństwa. Metodyka PTER umożliwia adaptację do postaci uszczegółowienia dziedzinowego dla metodyki szerzej traktującej tematykę testów np. metodyki P-PEN.

SŁOWA KLUCZOWE: bezpieczeństwo aplikacji internetowych (WWW), testy bezpieczeństwa, błędy bezpieczeństwa, metodyka testów bezpieczeństwa, testy penetracyjne

1. Wprowadzenie

Jak się okazuje, testy bezpieczeństwa są elementem często pomijanym podczas cyklu życia oprogramowania. Z powodów związanych głównie z wymaganiami biznesowymi często stosowanym podejściem staje się realizacja prac wdrożeniowych w jak najszybszym czasie i za jak najmniejszą cenę. Takie praktyki najczęściej nie uwzględniają problemów bezpieczeństwa, bądź traktują je jako problemy o niższym znaczeniu. Są to pozorne oszczędności, ponieważ w dalszej perspektywie nierzadko prowadzi to do sytuacji, w których nieprawidłowo zaprojektowane i/lub zaimplementowane aplikacje są przyczyną dużych strat nie tylko finansowych.

Podstawowym źródłem wiedzy dla zainteresowanych zagadnieniami testów bezpieczeństwa są strony internetowe i czasopisma branżowe, jednak przekazywane w nich informacje są w dużym stopniu wyrywkowe

i nieuporządkowane. Natomiast sami zainteresowani np. specjalistyczne firmy, pentesterzy, działy bezpieczeństwa w różnych organizacjach, wypracowują własne rozwiązania zależne od posiadanej wiedzy, jak i potrzeb. W analogicznej sytuacji znajdują się studenci informatyki o specjalnościach typu bezpieczeństwo systemów IT, którzy po ukończeniu studiów planują pracę w branży bezpieczeństwa.

Mając na uwadze wąską dziedzinę wiedzy, jaką są testy bezpieczeństwa współczesnych aplikacji WWW i zarazem ilość wiedzy niezbędnej do prawidłowego (miarodajnego) ich przeprowadzenia, popularyzacja metod realizacji przedmiotowych testów jest jak najbardziej pożądana. Podążając za P. Hope i B. Walther „*Niezależnie od stosowanych metodologii dotyczących jakości lub projektowania uwzględnienie bezpieczeństwa w planie testowania wymaga nowego do niego podejścia. Potrzebne są również specjalistyczne narzędzia umożliwiające testowanie zabezpieczeń*" [[4]].

Dlatego ważnym zadaniem wydaje się być stworzenie spójnej metodyki testów bezpieczeństwa aplikacji internetowych, która uwzględniałaby sytuacje kompromisowe związane z wymaganiami biznesowymi, pozwalając na jednoczesne utrzymanie odpowiedniego poziomu bezpieczeństwa. Metodyka testów powinna również przekrojowo wskazywać główne kroki niezbędne do ich realizacji, w sposób uniwersalny wskazując narzędzia i sposoby ich realizacji.

Opracowanie spójnej metodyki testów bezpieczeństwa wymaga sformułowania jasnych zasad oraz określenia kompleksowej procedury postępowania.

2. Założenia metodyki testów bezpieczeństwa aplikacji internetowych

W celu opracowania spójnej metodyki kompleksowych testów bezpieczeństwa aplikacji internetowych wymagane jest zrozumienie zarówno powodów przeprowadzania takich testów, jak i problemów związanych z ich realizacją. Powody przeprowadzania testów bezpieczeństwa aplikacji internetowych zostały dokładnie omówione w [1], zaś dla przypomnienia zostały poniżej tylko wymienione:

- spełnienie regulacji prawnych,
- spełnienie wymagań bezpieczeństwa,
- dopuszczenie aplikacji do użytku.

Jeśli zaś mówimy o metodyce takich testów, to można stwierdzić, że jej

celem nadrzędnym jest wykazanie poziomu uzasadnionego zaufania do aplikacji internetowej w wyniku przeprowadzonego procesu testowego oraz za pomocą udostępnionych narzędzi, w sposób przedstawiający wartość dla kadry zarządzającej i pozwalający na gromadzenie wykazanych podatności w celu późniejszego śledzenia postępów związanych z minimalizacją ryzyka.

Analizując dokładniej cel opracowania wytycznych jak przeprowadzać testy bezpieczeństwa aplikacji internetowych można powiedzieć, że kompleksowa metodyka powinna:

- dostarczyć wiedzy, jak należy wykonać testy (w szczególności tzw. „ścieżki technicznej"), w jaki sposób przeprowadzać je skutecznie i miarodajnie, jak zmniejszać ich koszty (np. wykorzystując darmowe rozwiązania),
- prowadzić do opracowania finalnego raportu w formie czytelnej, uwzględniającej zarówno wykazanie błędów wpływających na bezpieczeństwo aplikacji WWW, jak i potencjalnych możliwości ich wykorzystania w postaci scenariusza użycia oraz wskazania ewentualnych sposobów eliminacji, bądź minimalizacji powstałych na ich skutek ryzyk,
- pomagać w przezwyciężaniu problemów, które towarzyszą przedmiotowym testom (problemy te zostały dokładniej opisane w dalszej części tego punktu),
- dostarczyć wskazówek w jaki sposób zbudować repozytorium testów, w którym będą przechowywane zarówno wyniki poszczególnych testów, jak i statusy realizacji związane z naprawą błędów (usuwaniem podatności), bądź minimalizacją ryzyka,
- pobudzać świadomość związaną z testami bezpieczeństwa u zainteresowanych stron.

Problemy związane z realizacją testów bezpieczeństwa aplikacji internetowych mogą być spowodowane między innymi brakiem:

- zrozumienia istoty takich testów,
- odpowiedniej wiedzy merytorycznej związanej z ich wykonaniem,
- wystarczającej ilości czasu lub odpowiednich zasobów (finansowych oraz osobowych),
- priorytetyzacji wykrytych błędów i niezrozumiałym ich omówieniem skierowanym do strony testowanej,
- możliwości śledzenia postępów związanych z naprawą błędów, bądź zmniejszeniem ryzyka ich wystąpienia.

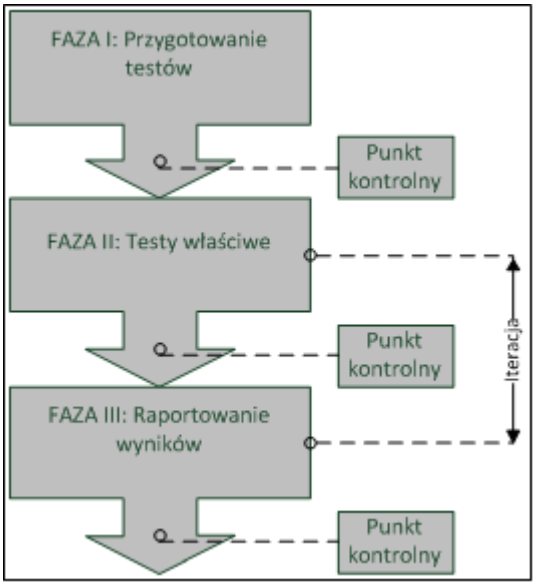
Przedstawione powyżej problemy zostały uwzględnione przy definiowaniu celów oraz założeń konstruowanej metodyki.

Opracowanie spójnej metodyki testów bezpieczeństwa aplikacji WWW jest istotnym elementem, który może wpłynąć na poziom ochrony zasobów informacyjnych. Należy mieć jednak na uwadze, że rzetelne testy bezpieczeństwa muszą być skoncentrowane na zapotrzebowaniu odbiorcy testów a zastosowane podejście metodologiczne musi być traktowane jako zestaw podstawowych i obowiązkowych czynności, uzupełnionych o niuanse związane właśnie z tymi potrzebami i dodatkową wiedzą ekspercką.

Założenia proponowanej metodyki

Ogólne założenia proponowanej metodyki obejmują 9 punktów określających:

1. **Sposób realizacji testów** - wskazujący między innymi kolejność wykonywanych czynności, z jednoczesnym pozostawieniem pewnej dowolności w wyborze sposobu ich realizacji np. w zakresie narzędzi, które warto wykorzystać w ich realizacji.
2. **Rodzaj stosowanej metody testowej** – dla poszczególnych etapów testów określono zalecane metody testowe (*gray box, white box*), z naciskiem na testy *gray box,* jako najbardziej odzwierciedlające rzeczywiste zagrożenia aplikacji internetowych. Celowo pominięto testy typu „*black box*", zwane również testami z „wiedzą zerową", ponieważ w praktyce realizacja rzetelnych testów bezpieczeństwa powinna podpierać się chociaż cząstkowymi informacjami na temat testowanej aplikacji (jak np. przeznaczenie biznesowe aplikacji, technologia jej wykonania, adres URL testowanej aplikacji itp.). Napastnicy najczęściej, przed przystąpieniem do praktycznego ataku, również prowadzą rozpoznanie wybranego celu, co przekłada się później na odpowiedni dobór scenariuszy testowych.
3. **Liczba faz metodyki** - jak pokazano na rys.1, metodyka składa się z trzech faz (nie uwzględniając w nich retestów). Zakłada się również realizację trzech punktów kontrolnych na koniec każdej z faz oraz realizację podczas fazy III iteracji (częściowej) kroków związanych z wykrywaniem podatności zidentyfikowanych w fazie II. Iteracja ma na celu prawidłowe oznaczenie wykrytych błędów w celu eliminacji błędów typu *false positive*. Fazy są podzielone na kroki. Wykonanie każdego kroku powinno dać konkretne informacje zwiększające wiedzę testujących na temat badanego systemu.



Rys. 1. Fazy projektowanej metodyki bezpieczeństwa aplikacji WWW

Punkty kontrolne dotyczą weryfikacji prac wykonanych w danej fazie na zasadzie weryfikacji i dokumentowania/rejestracji częściowych wyników testów.

W istocie wymienione fazy koncepcyjnie pokrywają się z metodyką testów bezpieczeństwa P-PEN: „Główna idea metodyki P-PEN polega na przeprowadzeniu prac w trzech etapach: analizy, właściwych badań i syntezy (dalej nazywanej integracją wyników)." [6].

Dodatkowo przy określaniu ilości faz należy podjąć decyzję, czy oprócz, w tym przypadku obligatoryjnej, ścieżki technicznej będzie realizowana ścieżka formalna.

4. **Specjalistyczne narzędzia -** wskazanie narzędzi służących do realizacji testów zgodnie z przyjętą metodyką oraz dostarczenia środowiska, w którym będą one dostępne dla testującego. Dodatkowo do celów związanych z punktami kontrolnymi zalecane jest udostępnienie repozytorium testów. Dla pojedynczego kroku w fazach I i II powinno zostać przedstawione co najmniej jedno narzędzie służące do jego przeprowadzania. Narzędzia stosowane w iteracji w fazie III powinny natomiast być najczęściej różne od użytych wcześniej.
5. **Możliwość powtórzeń wyników eksperymentów** - przy założeniu identycznych warunków testowych: środowiska, przypadków testowych. Możliwość identyfikacji zbliżonej liczby podatności.
6. **Formę raportu z testów -** zawierającego dodatkową część

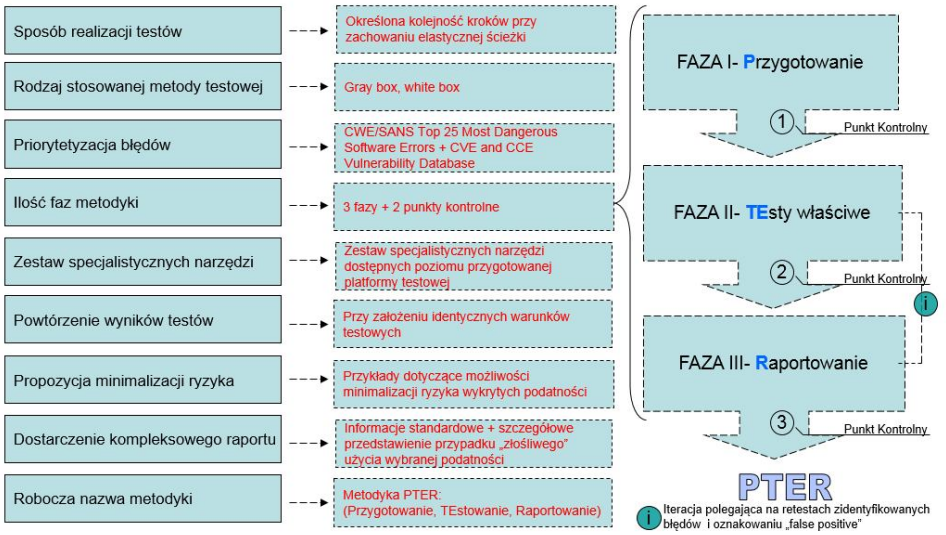
z podsumowaniem dla biznesu/kierownictwa oraz szczegółowym opisem przypadku użycia dla wybranych podatności o najwyższym poziomie ryzyka. Każda podatność powinna zawierać opis zawierający: liczbę porządkową, nazwę, numer CVE/CCE w przypadku błędów znanych, kategoria błędu w przypadku błędu nowego, poziom ryzyka, numer fazy i kroku, w którym została zidentyfikowana, informacje o ewentualnym powiązaniu z inną podatnością.

7. **Sposób klasyfikacji i szacowania poziomu istotności błędów bezpieczeństwa (ranking, *scoring*)** - element niezbędny do określenia prawidłowej kolejności realizacji właściwych środków zaradczych. Metodyka zakłada, że klasyfikacja zostanie opracowana:
   - na podstawie „2011CWE/SANS Top 25 Most Dangerous Software Errors" [21] dla błędów ogólnych,
   - na podstawie „CVE and CCE Vulnerability Database" [24] dla błędów o znanych numerach CVE/CCE,
   - zaś na podstawie publikacji [20] zostanie opracowania baza słownikowa używanych definicji związanych z podatnościami.
   - Klasyfikacja podatności, przy uwzględnieniu przykładowych przypadków użycia (w tym ich prawdopodobieństwa wystąpienia) oraz potencjalnych szkód tym spowodowanych stanowi podstawę do ostatecznego oszacowania poziomu istotności wykrytej podatności.
8. **Wskazanie sposobów minimalizacji ryzyka -** dostarczenie informacji związanych z możliwością minimalizacji skutków wykrytych zagrożeń.
9. **Roboczą nazwę metodyki testów bezp. aplikacji WWW -** Metodyka PTER- ***P**rzygotowanie, **TE**stowanie **R**aportowanie* (ang. ***P**reparing **TE**sting **R**eporting,* nazwa od początkowych liter faz metodyki). Logo metodyki zostało przedstawione na rysunku 2.

Rys. 2. Logo metodyki PTER

W celu łatwiejszej identyfikacji metodyki oraz zapamiętania jej nazwy, logo przedstawia **pter**odaktyla (gr. *Pterodactylus*)[1].

Rysunek 3 przedstawia zestawienie celów i założeń metodyki PTER.



Rys. 3. Zestawienie celów i założeń metodyki PTER

Ograniczenia tworzonej metodyki

Należy wyraźnie zaznaczyć, iż metodyka (PTER) testów bezpieczeństwa aplikacji WWW nie dotyczy pojęcia znacznie szerszego, jakim jest audyt informatyczny, a może stanowić jedynie jego uzupełnienie. Znaczenie użytego określenia „testy bezpieczeństwa" jest bliższe nazwie testy penetracyjne zdefiniowanej w Biuletynie Instytutu Automatyki i Robotyki Wojskowej Akademii Technicznej, przez dr. inż. A. E. Patkowskiego [6].

Zgodnie ze swoim przeznaczeniem metodyka PTER pomija w swoim zakresie realizację zagadnień związanych z lokalnymi testami bezpieczeństwa infrastruktury (testy konfiguracji systemów operacyjnych serwerów: HTTP, middleware[2], aplikacyjnych oraz SZBD[3], infrastruktury sieciowej itp.).

[1] Wikipedia, http://pl.wikipedia.org/wiki/Pterodaktyl
[2] Middleware- oprogramowanie pośredniczące
[3] SZBD- system zarządzania bazą danych

Oczywiście w celu realizacji kompleksowych testów infrastruktury teleinformatycznej, zalecane jest ich odrębne wykonanie, przykładowo za pomocą wspomnianej metodyki P-PEN.

W ramach metodyki PTER nie ma zdefiniowanej liczebności zespołu testowego, ważne jest natomiast, aby tester był ekspertem dziedzinowym w zakresie bezpieczeństwa aplikacji internetowych.

Dodatkowo ścieżka formalna przedstawiona w tabelach opisujących proces testowy jest zamieszczona tylko jako przykład, obrazujący możliwość uzupełnienia ścieżki technicznej o wymagania biznesowe.

3. Opis metodyki PTER testów bezpieczeństwa aplikacji internetowych

Ppunkt zawiera informacje związane ze ścieżką techniczną metodyki PTER. Podzielony jest on na trzy części – pierwszą przedstawiającą krótki opis przyjętej terminologii, drugą – definiującą proces realizacji metodyki w postaci obligatoryjnych do wykonania kroków i punktów kontrolnych oraz trzecią opisującą dodatkowe wymagania odnoszące się np. do formatu notatki po wykryciu podatności do natychmiastowego zgłoszenia.

3.1.1. Stosowana terminologia

Banner Grabbing – technika enumeracji informacji na temat wersji testowanego oprogramowania.

Code Injection – atak na aplikację WWW polegający na wykorzystaniu błędów związanych z możliwością wstrzyknięcia do aplikacji kodu, realizującego zamiary strony atakującej.

Cookie – mechanizm stosowany w technologiach internetowych, wprowadzony w celu uzupełnienia ograniczenia protokółu HTTP związanego z jego bezstanowością.

Cross-site Scripting – atak na aplikację WWW polegający na wykonaniu nieuprawnionego skryptu na zasobach zaufanej witryny.

Cross-site Request Forgery – atak na aplikację WWW polegający na wykonaniu przez uwierzytelnionego użytkownika żądania spreparowanego przez atakującego.

DoS – ang. *Denial of Service*, „atak typu odmowa usługi, nazywany też blokowaniem usług, jest działaniem mającym na celu wyczerpanie zasobów sieci (pochłanianie przepustowości) lub systemu komputerowego (czasu

procesora, pamięci, miejsca na dysku, procesów systemowych)" [5].

Exploit – „rutynowy algorytm (często w postaci skryptu lub programu) wykorzystania podatności systemu komputerowego na szkodę jego bezpieczeństwa" [6].

False positive – błąd sklasyfikowany jako prawdziwy, a w rzeczywistości będący błędem bez możliwości jego wykorzystania.

Fingerprinting – technika mająca na celu identyfikację systemu operacyjnego, aplikacji itp.

Fuzzer – narzędzie służące do automatyzowania wyszukiwania podatności aplikacji (bądź serwera) poprzez przekazywanie losowych danych. Stosowane do wykrywania podatności typu: przepełnienie bufora, *DoS, XSS, CSRF, SQL Injection, XPATH injection* itp.

Incorrect Logic flows – błędy w aplikacjach WWW związane z niepoprawnym przepływem logicznym.

LDAP Injection – atak na aplikację WWW typu *code injection* polegający na wykorzystywaniu błędów w konstrukcjach składniowych LDAP.

Misconfiguration – błędy w aplikacjach WWW związane z błędami w konfiguracji aplikacji (listing zbędnych katalogów itp.).

Narzędzia typu spider, crawler – narzędzia służące do zbierania informacji o zasobach stron WWW, najczęściej poprzez rekurencyjne przeszukiwanie sieci Internet.

Payload – ładunek zawierający listę potencjalnych wektorów ataków stosowany przy narzędziach typu *fuzzer*.

Pentester – osoba wykonująca testy penetracyjne.

Platforma testowa PTER – system operacyjny i oprogramowanie (użytkowe, specjalizowane, repozytorium testowe) zaprojektowane w celu realizacji metodyki PTER.

Repozytorium testowe – oprogramowanie pozwalające na przechowywanie dokumentów z testów w postaci notatek, raportów oraz zrzutów ekranu (tzw. artefaktów testowych).

Rekonesans aktywny – pozyskiwanie informacji na temat badanego elementu za pomocą bezpośredniej interakcji (np. próbkowanie, skanowanie sieciowe).

Rekonesans pasywny – pozyskiwanie informacji na temat badanego elementu w sposób niewymagający bezpośredniej interakcji (np. z ogólnodostępnych źródeł, bazy Whois).

Session Fixation – atak na aplikację WWW związany z przechwyceniem sesji innego użytkownika.

Session Poisoning – atak na aplikację WWW związany z możliwością modyfikacji danych znajdujących się w sesji użytkownika.

SQL Injection – błąd typu *code injection* dotyczący manipulowania zapytaniami SQL.

System CMS – ang. *Content Management System,* system zarządzania treścią.

True positive – błąd prawdziwy sklasyfikowany prawidłowo (błąd z potwierdzoną możliwością jego użycia).

UML –ang. *Unified Modeling Language*, zunifikowany język modelowania.

Webservices – usługi internetowe, zdefiniowane za pomocą języka opisu usług – standaryzowanym językiem, bazującym na XML.

XPath Injection – atak na aplikację WWW typu *code injection* polegający na zastosowaniu manipulacji w wyszukiwaniu *xpath*, celem wydobycia informacji z baz danych XML.

3.1.2. Opis kolejnych faz metodyki PTER

W poniższym podpunkcie zostały opisane najważniejsze elementy metodyki PTER, które wskazują w jaki sposób odpowiednio przeprowadzić kolejne fazy testowania.

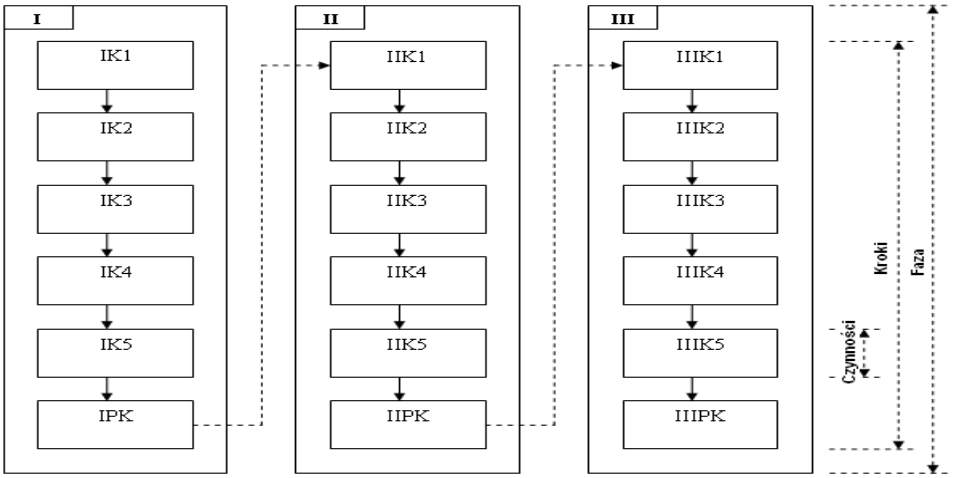
Zgodnie z przyjętymi założeniami metodyka składa się z 3 faz:

- Fazy I - przygotowania,
- Fazy II - testów właściwych,
- Fazy III - raportowania wyników,

oraz 3 punktów kontrolnych realizowanych na koniec fazy I, II oraz III. Dla każdej fazy został przedstawiony krótki opis dotyczący:

- zakresu i ograniczeń proponowanej metody badania w odniesieniu do danej fazy testów,
- przyjętej procedury badań,
- specyfikacji typów proponowanych narzędzi,
- planu badań i przewidywanego zakresu czasowego,
- zastosowanego podejścia w celu zapewnienia o prawdziwości i rzetelności wyników.

Rysunek 4 ilustruje mapę struktury metodyki zgodnie z przyjętymi oznaczeniami dla faz, kroków i punktów kontrolnych.



Rys. 4. Oznaczenia przyjętych skrótów przedstawione na rysunku struktury metodyki

Faza I- Przygotowanie

W fazie przygotowania testujący realizuje czynności związane ze zebraniem dokumentacji technicznej i informacji dotyczących wymagań niefunkcjonalnych związanych z bezpieczeństwem aplikacji, jak również odwzorowania struktury aplikacji oraz wstępnego rekonesansu technicznego testowanej infrastruktury. Określane są również punkty wejściowe do aplikacji. Zebrane informacje zostaną wykorzystane w kolejnej fazie testów.

Opis zakresu i ograniczeń proponowanej metody badań w odniesieniu do fazy I testów

Zakres fazy „Przygotowanie" obejmuje realizację czynności mających na celu udostępnienie odpowiedniego planu testów, według którego powinna odbywać się kolejna faza testów właściwych oraz dobór narzędzi pozwalających na ich wykonanie.

Przedstawiona w opisie procedury opcjonalna ścieżka formalna jest tylko propozycją i nie będzie omawiana w niniejszym opracowaniu. Jej uwzględnienie zależy tylko od wymagań biznesowych przedstawionych w umowie z testującym i w żadnym przypadku nie stanowi obowiązkowego elementu.

Opis procedury badań

Na procedurę badań w fazie przygotowania składają się punkty wymienione w tabeli 1.

Tabela 1. Procedura badań PTER- faza I

PTER, faza I- *Przygotowanie*		
	Ścieżka techniczna	**Ścieżka formalna (opcjonalna)**
IK1	Pozyskanie podstawowych informacji technicznych na temat testowanej aplikacji. Specyfikacja adresów IP oraz URL wchodzących w skład testowanej infrastruktury.	Pozyskanie podstawowych informacji związanych z biznesowymi celami testowanej aplikacji.
IK2	Wykonanie pasywnego i aktywnego rekonesansu dostępnych usług (skanowanie portów sieciowych).	Pozyskanie dokumentacji zawierającej wymagania bezpieczeństwa dotyczące testowanej aplikacji oraz polityki bezp. Uzyskanie pisemnej zgody na wykonanie rekonesansu aktywnego.
IK3	Wykonanie pasywnego i aktywnego rekonesansu używanych technologii (wersje i rodzaj oprogramowania).	Uzyskanie pisemnej zgody na wykonanie testów właściwych w fazie II.
IK4	Odwzorowanie struktury aplikacji internetowej oraz określenie możliwych punktów wejściowych.	Pozyskanie informacji na temat możliwych uwarunkowań formalno-prawnych związanych z testowaną aplikacją (UoDo, PCI/DSS itp.).
IK5	Wykonanie (na bazie zebranych informacji) modelu działań związanych z testami właściwymi oraz ogólnego harmonogramu testów.	Sprawdzenie licencji dotyczących uzyskanej listy oprogramowania.
PTER, faza I- *punkt kontrolny*		
IPK	Zapisanie wyników w repozytorium testowym.	Udokumentowanie przeprowadzonych działań.

Kolejne kroki wchodzące w skład fazy I zostały opisane poniżej:

Krok 1 (IK1) - pozyskanie podstawowych informacji technicznych oraz specyfikacja adresów IP i URL wchodzących w skład testowanej infrastruktury.

Cel: w kroku należy zapoznać się z dokumentacją techniczną oraz dokonać przedstawienia adresów URL i dostępnego zakresu adresacji IP wchodzącej w skład testowanej infrastruktury aplikacji WWW.

Wejście: dokumentacja techniczna, adres URL i IP aplikacji.

Wyjście: wyspecyfikowane adresy URL oraz IP.

Uzasadnienie: zapoznanie się z testowanym obiektem, przygotowanie infrastruktury testowej, odpowiedni dobór narzędzi i próbek testowych.

Krok 2 (IK2) - specyfikacja dostępnych portów sieciowych.

Cel: krok ma na celu specyfikację dostępnych portów sieciowych

testowanej infrastruktury.

Wejście: adresy IP z kroku 1.

Wyjście: wyspecyfikowane numery portów dostępnych zdalnie.

Uzasadnienie: przygotowanie infrastruktury testowej, budowa mapy testowanej aplikacji WWW.

Krok 3 (IK3) - specyfikacja technologii.

Cel: aplikacje WWW to obecnie struktury hybrydowe, złożone z różnych komponentów technologicznych. Krok ma na celu stworzenie prawdopodobnego zbioru zastosowanych w testowanej infrastrukturze technologii za pomocą technik rekonesansu pasywnego i aktywnego.

Wejście: adresy IP i numery portów z wcześniejszych kroków, kody źródłowe.

Wyjście: specyfikacja rodzajów użytych technologii.

Uzasadnienie: przygotowanie infrastruktury testowej, odpowiedni dobór narzędzi i próbek testowych.

Krok 4 (IK4) - specyfikacja struktury aplikacji i punktów wejściowych.

Cel: bazą do realizacji kolejnej fazy jest poznanie struktury aplikacji, dostępnych stron, podstron oraz punktów wejściowych, na których powinny skupić się dalsze badania.

Wejście: adresy URL, pod którymi dostępna jest testowana aplikacja.

Wyjście: odwzorowanie struktury aplikacji w postaci listy stron i ich hierarchii oraz specyfikacja punktów wejściowych w postaci: formularzy, pól wejściowych, użytego rodzaju kodowania, mechanizmu *cookie*, stosowanych metod protokołu HTTP itp.

Uzasadnienie: przygotowanie infrastruktury testowej, odpowiedni dobór narzędzi i próbek testowych, wsad do budowy modelu działań.

Krok 5 (IK5) - przygotowanie modelu działań i wstępnego harmonogramu.

Cel: zapewnienie możliwości realizacji działań w kolejnych fazach zgodnie z harmonogramem czasowym i według wstępnego planu.

Wejście: zbiory wynikowe ze wszystkich poprzednich kroków.

Wyjście: wstępny harmonogram z wyszczególnieniem działań w kolejnych fazach. Model działań może zostać przedstawiony na bazie konstrukcji diagramu przypadków użycia i scenariuszy działań.

Uzasadnienie: przygotowanie infrastruktury testowej, przygotowanie modelu czynności testowych.

Punkt kontrolny (IPK) - zapisanie zebranych informacji w repozytorium.

Cel: udokumentowanie zrealizowanych działań, wsad do raportu.

Wejście: zbiory wynikowe ze wszystkich kolejnych kroków.

Wyjście: odpowiednie zapisy w repozytorium testowym.

Uzasadnienie: wszystkie zebrane w fazie i informacje powinny być przechowane w repozytorium o ustalonej strukturze, w celu zapewnienia łatwego dostępu dla testującego lub innych testujących (jeśli testy wykonuje kilka osób) podczas realizacji kolejnych faz.

Specyfikacja proponowanych narzędzi

Poniżej przedstawiono ogólną listę typów narzędzi z przypisaniem ich do poszczególnych kroków w fazie I. Celowo nie wskazywano konkretnych narzędzi i programów, ponieważ każdy tester ma swoje preferencje, a dodatkowo rotacja narzędzi jest dość duża z racji tego, że niektóre z nich nie są rozwijane i dostosowywane do zmieniających się potrzeb, a w ich miejsce pojawiają się nowe.

Krok 1:

- repozytorium testowe,
- edytor biurowy.

Krok 2:

- skaner adresów IP,
- skaner sieciowy,

Krok 3:

- narzędzia do wykrywania technologii typu *banner grabbing* i *fingerprinting,*
- monitor żądań/odpowiedzi HTTP,
- narzędzia rekonesansu pasywnego: skanowanie kodu, usługi online, inne narzędzia,
- narzędzia rekonesansu aktywnego,

Krok 4:

- narzędzie do tworzenia lokalnej kopii (mirroru) aplikacji,
- narzędzia typu *spider*/*crawler*,
- narzędzie do wyszukiwania punktów wejściowych, np. formularzy HTML, parametrów URL, nagłówków i żądań protokołu HTTP, ciasteczek, parametrów sesji itp.

Krok 5:

- narzędzie do wspomagania procesu harmonogramowania,

- narzędzie do tworzenia modelu działań w notacji UML.

Punkt kontrolny:

- repozytorium testowe,
- edytor biurowy.

Na osobną uwagę zasługuje kwestia repozytorium testowego. Jest to element niezwykle ważny, a jednocześnie, jak się okazuje, rzadko przy tego typu pracach wykorzystuje się do przechowywania wyników badań, w szczególności cząstkowych odpowiednio przygotowane repozytoria. Jak przedstawiono w tabeli 6 podsumowującej dostępne metodyki [1], żadna z nich nie dostarcza gotowego do użycia repozytorium, bądź nie przedstawia założeń co do jego zaprojektowania. Metodyka PTER zakłada dostarczenie gotowego repozytorium przechowującego dane testowe, które pozwalają testerowi m.in. na zarządzanie przebiegiem testów oraz cyklem życia wykrytych podatności w przypadku testów cyklicznych. Struktura i forma przechowywania wyników badań ma duży wpływ na ich późniejszą analizę i w konsekwencji na końcowe wnioski. Czasami niezauważenie określonego faktu wynikającego z wcześniejszych badań może doprowadzić do pominięcia w dalszych testach istotnego elementu, który posiada podatności. We wspomnianym repozytorium powinny być również przechowywane statusy realizacji kolejnych czynności testowych oraz, po zakończeniu testów statusy realizacji czynności naprawczych. Z tego też względu metodyka PTER kładzie duży nacisk na stosowanie dedykowanego celom testów penetracyjnych repozytorium i w ramach jej opisu zostało ono wyspecyfikowane. W metodyce PTER założono, że repozytorium będzie:

- dostępne przez przeglądarkę internetową, dzięki czemu będzie pozwalało na równoczesny dostęp kilku osobom;
- pozwalało na dostosowanie do potrzeb metodyki w taki sposób, aby możliwe było gromadzenie danych dotyczących testów, przypisywanie poziomów ryzyka związanych z podatnościami oraz śledzenie postępów związanych z ich likwidacją;
- umożliwiało odnotowanie w oddzielnych polach (co ułatwi późniejsze wyszukiwanie, filtrowanie zawartości repozytorium w oparciu o te atrybuty), między innymi, informacji nt. sygnatury podatności, jej nazwy, opisu, istotności (ryzyka), statusu weryfikacji pod *kątem true/false positive*, zalecenia, priorytetu naprawy, stanu naprawy;
- umożliwiało szybkie wykonywanie kopii bezpieczeństwa.

W proponowanej implementacji repozytorium testowe bazuje na systemie Redmine, przeznaczonym do celów zarządzania projektami informatycznymi.

Opis planu testów i zakresu czasowego

Podczas realizacji fazy I można wyróżnić następujące kamienie milowe: spotkanie ze stroną zlecającą związane z pozyskaniem dokumentacji i informacji uzupełniających, rekonesans techniczny i odtworzenie struktury aplikacji, określenie punktów wejściowych oraz budowa harmonogramu testów z przedstawieniem działań do realizacji w fazie II.

Plan testów realizowanych w fazie I powinien odzwierciedlać, co najmniej, kroki związane z procedurą testową i przypisanymi do nich: orientacyjnym czasem trwania oraz przewidywanym czasem rozpoczęcia i zakończenia.

Stworzony plan testów powinien być udostępniany na życzenie strony zlecającej testy. Zlecający może zażądać przedstawienia takiego planu podczas spotkania inicjującego testy, realizowanego zazwyczaj w gronie administratorów technicznych odpowiedzialnych za badane aplikacje aby lepiej zrozumieć ich zakres oraz wskazać termin badań poszczególnych elementów testowanego środowiska. Służby techniczne zlecającego mogą również mieć potrzebę potwierdzenia wykonania testów po stronie badanej infrastruktury, np. poprzez weryfikację zapisów w odpowiednich dziennikach zdarzeń (tzw. logów).

Dodatkowo, ze strony zespołu testowego plan testów służy do realizacji celów związanych z samoorganizacją zespołu. Stworzony plan wprowadza możliwość nadzorowania testów poprzez wprowadzenie przedziałów czasowych oraz obrazuje zakres metodyki poprzez opis wykonywanych czynności. Przykładowy plan przedstawiono w tabeli 2.

Tabela 2. Przykład planu testów PTER- faza I

Krok	Opis	Data od	Data do	Przewid. czas [h]
1	Spotkanie ze stroną zlecającą	01.09.2015	03.09.2015	5
2	Rekonesans techniczny	05.09.2015	06.09.2015	3
3	Odtworzenie struktury aplikacji	05.09.2015	06.09.2015	3
4	Określenie punktów wejściowych	06.09.2015	07.09.2015	5
5	Wstępny projekt harmonogramu	07.09.2015	07.09.2015	2
			Suma:	**18 [h]**

Opis zastosowanego podejścia w celu zapewnienia o prawdziwości i rzetelności wyników

Jeśli testujący posiada dostarczoną przez zlecającego dokumentację techniczną, w fazie tej powinien potwierdzić wykryte adresy, porty i technologie odwołując się do jej konkretnych punktów w sporządzonej notatce. Notatka powinna zawierać również wszystkie informacje związane z wykonanymi czynnościami oraz ich wynikami. Każda wykryta podatność powinna być zilustrowana zrzutem ekranu lub nagranym filmem (o ile to technicznie możliwe) wraz ze stosownym opisem związanym z jej wykryciem.

Faza II- Testy właściwe

Opis zakresu i ograniczeń proponowanej metody badań w odniesieniu do fazy II testów

Faza „Testy właściwe" powinna obejmować realizację czynności związanych z testami penetracyjnymi opisanymi w poszczególnych krokach procedury testowej.

Należy zauważyć, że ze względu na specyfikę takich testów, nie można narzucać szczegółowych sposobów postępowania, ponieważ mogą być one różne, chociażby ze względu na heterogeniczność aplikacji WWW. Dlatego też przyjęta metoda opisuje kroki jako punkty obligatoryjne jednak realizacja zawartych w nich czynności jest elastyczna i zależy od specyfiki testowanej aplikacji. Należy również mieć na uwadze, iż równocześnie z rozwojem aplikacji internetowych zmienia się charakter ich testów, stąd krok realizujący zbiór testów dodatkowych jest miejscem mającym na celu uzupełnienie fazy o nowości związane z tematyką.

Analogicznie do poprzedniego punktu, przedstawiona w opisie procedury opcjonalna ścieżka formalna jest tylko propozycją i nie będzie omawiana w niniejszym opracowaniu.

Opis procedury badań

Na procedurę badań w fazie przygotowania składają się punkty wymienione w przedstawionej tabeli nr 3.

Tabela 3. Procedura badań PTER- faza II

PTER, faza II- *Testy właściwe*		
	Ścieżka techniczna	**Ścieżka formalna (opcjonalna)**
IIK1	Wykonanie testów: uwierzytelniania, autoryzacji oraz mechanizmów zarządzania sesją aplikacji.	Wykonanie analizy ewentualnych dokumentacji dotyczących mechanizmów uwierzytelniania, autoryzacji oraz zarządzania sesją.
IIK2	Wykonanie testów konfiguracji aplikacji, sprawdzenie błędów generowanych przez aplikację i jej komponenty oraz wykonanie testów mających na celu wykrycie podatności.	Wykonanie analizy ewentualnych dokumentacji dotyczących konfiguracji aplikacji.
IIK3	Wykonanie testów walidacji danych wejściowych w punktach określonych podczas realizacji kroku 4 fazy I (IK4).	Uzyskanie pisemnej zgody na wykonanie opcjonalnych destrukcyjnych testów właściwych (IIK5).
IIK4	Wykonanie testów logiki biznesowej aplikacji- jako opcja po uzgodnieniach ze stroną zlecającą.	Wykonanie analizy ewentualnych dokumentacji związanych z logiką biznesową aplikacji oraz przetwarzanymi w niej informacjami (celem przygotowania specyficznych scenariuszy testowych).
IIK5	Wykonanie testów dodatkowych (Web Services, test CMS, SSL itp).	Wykonanie lub weryfikacja analizy ryzyka związanej z testowaną aplikacją.
PTER, faza II- *punkt kontrolny*		
IIPK	Zapisanie wyników w repozytorium testowym.	Udokumentowanie przeprowadzonych działań.

Kolejne kroki wchodzące w skład fazy II zostały opisane poniżej:

Krok 1 (IIK1) - testy mechanizmów uwierzytelniania, autoryzacji i zarządzania sesją.

Cel: sprawdzenie poprawności działania mechanizmów uwierzytelniania, autoryzacji użytkowników oraz zarządzania sesją.

Wejście: formatki logowania, mechanizm *cookie*, konto użytkownika, identyfikatory sesyjne.

Wyjście: opis przeprowadzonych badań, ewentualna lista błędów związanych z wymienionymi mechanizmami (np. niepoprawne zarządzanie sesją: *Session Fixation, Session Poisoning*).

Uzasadnienie: wykazanie racjonalnie uzasadnionego poziomu zaufania do stosowanych mechanizmów: uwierzytelniania, autoryzacji i zarządzania sesją.

Krok 2 (IIK2) - testy konfiguracji oraz podatności aplikacji.

Cel: sprawdzenie poprawności konfiguracji aplikacji WWW (np. dostęp do stron administracyjnych) oraz poszukiwanie podatności komponentów aplikacji.

Wejście: struktura aplikacji, aktualna baza podatności.

Wyjście: opis przeprowadzonych badań, ewentualna lista błędów związanych z wymienionymi czynnikami (np. błędy typu *misconfiguration*).

Uzasadnienie: wykazanie racjonalnie uzasadnionego poziomu zaufania w obszarach: konfiguracji aplikacji oraz braku podatności komponentów aplikacji.

Krok 3 (IIK3) - testy walidacji danych wejściowych.

Cel: znalezienie ewentualnych błędów związanych z manipulacją danymi wejściowymi, które mogą wpływać na atrybuty bezpieczeństwa aplikacji WWW.

Wejście: lista punktów wejściowych, specyficznych atrybutów elementów, baza tzw. *payloadów*, *exploitów*.

Wyjście: wskazanie błędów dotyczących zależności pomiędzy danymi wejściowymi, a odpowiedzią aplikacji (np. błędy typu: *Cross-site Scripting* (XSS), *Cross-site Request Forgery* (CSRF), *Code Injection*, *SQL Injection*, *LDAP Injection*, *XPath Injection* oraz niepożądanych wycieków danych i niepoprawnie obsługiwanych błędów).

Uzasadnienie: wykazanie racjonalnie uzasadnionego poziomu zaufania w obszarach: walidacji danych wejściowych, braku wycieków informacji wrażliwych związanych z generowanymi błędami.

Krok 4 (IIK4) - testy logiki biznesowej.

Cel: przedstawienie możliwości działania na logikę biznesową testowanej aplikacji w taki sposób, aby za pomocą realizacji złośliwych czynności spowodować w rezultacie wykonania dozwolonych działań możliwość uzyskania nieuprawnionych korzyści, bądź wyrządzenia szkód. Stworzenie specyficznych scenariuszy testowych.

Wejście: struktura aplikacji, dokumentacja biznesowa, diagramy przypadków użycia.

Wyjście: wykazanie błędów związanych z logiką biznesową (np. błędy typu *incorrect logic flows*), zaktualizowane diagramy o czynności „złośliwego użycia".

Uzasadnienie: wykazanie racjonalnie uzasadnionego poziomu zaufania

w obszarach: braku błędów konstrukcji logicznej, niemożliwości wykorzystania prawidłowych, dostępnych akcji do złośliwych czynności.

Krok 5 (IIK5) -- testy dodatkowe.

Cel: uzupełnienie testów właściwych o testy specyficzne związane z wykrytymi podatnościami bądź obszarami uwzględnionymi w fazie I (np. wykryte technologie, wymagania zlecającego) oraz mechanizmami szyfrowania.

Wejście: lista wykrytych podatności, wykryte technologie z fazy I, strona działająca na protokole HTTPS.

Wyjście: błędy związane z rodzajem technologii (np. *Webservices*, systemy *CMS*), dodatkowymi wymaganiami zlecającego (np. realizacja zaleceń standardu PCI/DSS) oraz poziomem bezpieczeństwa użytych algorytmów szyfrujących.

Uzasadnienie: wykazanie racjonalnie uzasadnionego poziomu zaufania w obszarach: implementacji algorytmów szyfrujących, usług *Webservices*.

Punkt kontrolny (IIPK)- zapisanie zebranych informacji w repozytorium.

Cel: wszystkie zebrane w fazie II informacje powinny być przechowane w repozytorium o ustalonej strukturze, w celu zapewnienia łatwego dostępu dla testującego podczas realizacji kolejnych faz.

Wejście: zbiory wynikowe ze wszystkich kolejnych kroków.

Wyjście: odpowiednie zapisy w repozytorium testowym.

Uzasadnienie: wszystkie zebrane w fazie II informacje oraz zidentyfikowane błędy, powinny być przechowane w repozytorium o ustalonej strukturze, w celu zapewnienia łatwego dostępu dla testującego podczas realizacji kolejnej fazy.

Specyfikacja proponowanych narzędzi

Poniżej przedstawiono listę narzędzi z podziałem na kroki fazy II:

Krok 1:

- narzędzia do badania mechanizmów uwierzytelniania,
- narzędzia do badania mechanizmów autoryzacji,
- narzędzia do badania mechanizmów zarządzania sesją.

Krok 2:

- skaner CGI,
- skaner konfiguracji,
- skaner podatności WWW.

Krok 3:

- narzędzia typu fuzzer do celów: badania podatności typu Injection (XSS, SQLi, itp.),

Krok 4:

- testy manualne z użyciem narzędzi pomocniczych,
- narzędzia pomocnicze typu *Intercepting Proxy*.

Krok 5:

- narzędzia dedykowane do *Webservices*,
- narzędzia dedykowane do CMS,
- narzędzia dedykowane do szyfrowania SSL,
- bazy exploitów,
- narzędzia wspomagające użycie exploitów.

Punkt kontrolny:

- repozytorium testowe,
- edytor biurowy.

Opis planu testów i zakresu czasowego

Plan testów realizowanych w fazie II powinien przedstawiać wymagane kroki związane z procedurą testową i przypisanymi do nich orientacyjnym czasem trwania (w tym przewidywanym czasem rozpoczęcia i zakończenia). Główne kamienie milowe do realizacji w fazie II to:

- testy mechanizmów: uwierzytelniania, autoryzacji, zarządzania sesją,
- testy poprawności konfiguracji obiektu badań,
- testy walidacji danych wejściowych,
- testy logiki biznesowej,
- testy dodatkowe (zależą od specyfiki testowanej aplikacji, np. testy usług *Webservices*, systemu *CMS* bądź specyficznych ataków: *DoS*, algorytmów szyfrujących). Przykładowy plan przedstawiono w tabeli 4.

Opis zastosowanego podejścia w celu zapewnienia o prawdziwości i rzetelności wyników

Każdy z błędów powinien być zobrazowany o ile to technicznie możliwe zrzutem ekranu z jego wynikiem lub filmem ze stosownym opisem czynności związanych z jego wykonaniem. Dokumentacja powinna pozwolić na odtworzenie czynności testowych przez zlecającego (z wyjątkiem przypadków, w których użyte jest oprogramowanie będące własnością testującego, bądź za pomocą metod będących jego własnością intelektualną).

Tabela 4. Przykład planu testów PTER- faza II

Krok	Opis	Data od	Data do	Przewidywany czas [h]
1	Testy mechanizmów …	08.09.2015	09.09.2015	5
2	Testy pop. konfiguracji …	09.09.2015	10.09.2015	5
3	Testy walidacji danych	10.09.2015	11.09.2015	5
4	Testy logiki biznesowej	11.09.2015	12.09.2015	5
5	Testy dodatkowe	12.09.2015	13.09.2015	5
			Suma:	**25 [h]**

Faza III- Raportowanie

Opis zakresu i ograniczeń proponowanej metody badań w odniesieniu do fazy III testów

„Raportowanie" jest fazą obejmującą czynności związane zarówno z potwierdzeniem wykrytych podatności, jak i z odpowiednim ich oznakowaniem oraz przedstawieniem zlecającemu w postaci finalnego dokumentu. Podczas tej fazy określa się przypisanie błędów do odpowiedniej kategorii oraz dodaje się do nich odpowiednie poziomy ryzyka. Dodatkowo stworzona lista potwierdzonych błędów powinna posiadać zaproponowane sposoby minimalizacji bądź likwidacji ryzyka.

Każda opisana podatność powinna posiadać następującą strukturę:

- liczba porządkowa,
- nazwa,
- numer CVE/CCE w przypadku błędów znanych, kategoria błędu w przypadku błędu nowego,
- poziom ryzyka,
- numer fazy i kroku, w którym została zidentyfikowana,
- powiązanie z inną podatnością,
- zalecane środki zaradcze (zalecenia).

Do finalnego raportu powinien również zostać dodany (o ile jest to możliwe) szczegółowy opis techniczny wybranych błędów o najwyższym poziomie istotności. Czynność ta ma na celu pobudzenie świadomości i pomoc

w zrozumieniu istoty sytuacji przez stronę zlecającą. Mowa jest o wybranych podatnościach, ponieważ niektóre interpretacje błędów wymagają czynności nie zawsze akceptowanych przez stronę zlecającą, np. możliwości udostępnienia przeglądu kodu źródłowego, konfiguracji itp.

Dokładnie tak jak w poprzednich fazach przedstawiona w opisie procedury opcjonalna ścieżka formalna jest tylko propozycją i nie będzie omawiana w niniejszym opracowaniu.

Opis procedury badań

Faza raportowania realizowana jest zgodnie z punktami określonymi w tabeli 5.

Tabela 5. Procedura badań PTER- faza III

PTER, faza III- *Raportowanie*		
	Ścieżka techniczna	**Ścieżka formalna (opcjonalna)**
IIIK1	Weryfikacja wszystkich zidentyfikowanych i umieszczonych w repozytorium testowym błędów.	Wykonanie dokumentu opisowego (notatki) o stopniu pokrycia przedstawionej dokumentacji dotyczącej wymagań bezpieczeństwa.
IIIK2	Wykonanie ostatecznego podziału błędów na kategorie i przypisanie im poziomu ryzyka (wg. punktacji CVSS). Potwierdzenie wyników przeprowadzonych testów i oznaczenie potencjalnych błędów typu *false positive*.	Wykonanie dokumentu opisowego (notatki) o odniesieniu się do regulacji prawnych/polityk bezpieczeństwa.
IIIK3	Wykonanie dla wybranych błędów o najwyższym ryzyku przypadków użycia oraz szczegółowego opisu technicznego.	Wykonanie dokumentu opisowego (notatki) związanego z legalnością licencji zidentyfikowanego oprogramowania.
IIIK4	Stworzenie propozycji sposobów minimalizacji ryzyka wykorzystania zidentyfikowanych błędów.	Uzyskanie pisemnego potwierdzenia braku wpływu testów na działanie aplikacji.
IIIK5	Przekazanie raportu osobom zainteresowanym oraz organizacja spotkania podsumowującego.	
PTER, faza III- *punkt kontrolny*		
IIIPK	Zapisanie wyników w repozytorium testowym.	Zorganizowanie spotkania podsumowującego wyniki.

Kolejne kroki wchodzące w skład fazy III zostały opisane poniżej:

Krok 1 (IIIK1) - weryfikacja zidentyfikowanych błędów.

Cel: dostarczenie finalnej listy błędów jako wsadu do raportu z testów bezpieczeństwa, oznaczenie zweryfikowanych błędów jako błędy typu *true positive* (TP) oraz *false positive* (FP).

Wejście: lista błędów.

Wyjście: potwierdzona lista błędów z oznakowaniem na błędy typu TP lub FP oraz nadanym numerem porządkowym.

Uzasadnienie: specyfikacja przedstawionych błędów powinna zawierać prawidłowe oznaczenia błędów.

Krok 2 (IIIK2) - przypisanie do zweryfikowanych błędów poziomu istotności.

Cel: dostarczenie listy błędów z przypisanym poziomem ryzyka.

Wejście: potwierdzona lista błędów, aktualna punktacja CVSS[4], numery CWE.

Wyjście: lista błędów z przypisanym poziomem ryzyka.

Uzasadnienie: specyfikacja przedstawionych błędów powinna zawierać przypisane poziomy ryzyka pozwalające na odpowiednią ich priorytetyzację przez zleceniodawcę, np. w celu ustalenia kolejności usuwania.

Krok 3 (IIIK3) - stworzenie szczegółowego opisu wybranych błędów.

Cel: szczegółowe opisanie wybranych błędów o najwyższym poziomie ryzyka w celu przedstawienia ich w raporcie z testów.

Wejście: błędy o najwyższym poziomie ryzyka.

Wyjście: diagram przypadków użycia wybranych błędów, uzupełniony opisem technicznym związanym z możliwością ich wykorzystania.

Uzasadnienie: podniesienie poziomu wiedzy i zobrazowanie istotności zidentyfikowanych zagrożeń. Dostarczenie kadrze zarządzającej uzasadnienia dla ewentualnych nakładów finansowych.

Krok 4 (IIIK4) - przedstawienie sposobów minimalizacji ryzyka.

Cel: dostarczenie zlecającemu przykładów możliwych sposobów minimalizowania lub całkowitego usunięcia ryzyka dotyczącego poszczególnych błędów.

Wejście: potwierdzona lista błędów, aktualna punktacja CVSS.

Wyjście: lista błędów z przypisanym sposobem minimalizacji ryzyka.

Uzasadnienie: rzetelność przeprowadzonych testów i uzyskanych wyników powinna być podparta dostarczeniem dla zleceniodawcy przykładowych zaleceń naprawy błędów.

Krok 5 (IIIK5) - stworzenie raportu.

[4] Common Vulnerability Scoring System

Cel: sporządzenie raportu według wzoru.

Wejście: potwierdzona lista błędów, aktualna punktacja CVSS, sposoby minimalizacji ryzyka.

Wyjście: raport finalny.

Uzasadnienie: udokumentowanie prac testowych w formie powtarzalnej.

Punkt kontrolny (IIIPK) - zapisanie zebranych informacji w repozytorium.

Cel: zapisanie raportu w repozytorium testowym, organizacja spotkania podsumowującego.

Wejście: zbiory wynikowe ze wszystkich kolejnych kroków w fazie (raport).

Wyjście: odpowiednie zapisy w repozytorium testowymi i spotkanie podsumowujące.

Uzasadnienie: wszystkie zebrane w fazie III informacje oraz potwierdzone błędy powinny być przechowane w repozytorium o ustalonej strukturze, w celu zapewnienia łatwego dostępu dla testującego podczas realizacji prac naprawczych związanych z minimalizacją ryzyka.

Specyfikacja proponowanych zasobów

Poniżej przedstawiono listę narzędzi i pomocnych zasobów z podziałem na kroki fazy III:

Krok 1:

- testy manualne/automatyczne przy użyciu innych narzędzi niż w poprzednich krokach (do wyboru z listy dostępnych).

Krok 2:

- kalkulator NIST.

Krok 3:

- narzędzie do realizacji diagramu przypadków użycia w notacji UML,
- edytor biurowy.

Krok 4:

- doświadczenie własne, publikacje NIST.

Krok 5:

- edytor biurowy,
- bazy podatności CVE.

Punkt kontrolny:

- repozytorium testowe,

- edytor biurowy.

Opis planu testów i zakresu czasowego

Plan testów realizowanych w fazie III powinien przedstawiać minimalnie kroki związane z procedurą testową i przypisanymi do nich orientacyjnym czasem trwania i przewidywanym czasem rozpoczęcia i zakończenia. Główne kamienie milowe, które można wyodrębnić z proponowanej procedury w fazie Raportowania obejmują: weryfikację wykrytych podatności, podział i oznakowanie błędów, szczegółowy opis wybranych przypadków użycia podatności, przedstawienie sposobu minimalizacji bądź likwidacji ryzyka, stworzenie i przekazanie raportu stronie zlecającej. Przykład harmonogramu dla fazy III został przedstawiony w tabeli 6.

Opis zastosowanego podejścia w celu zapewnienia o prawdziwości i rzetelności wyników

Lista potencjalnych błędów wykrytych w fazie II podlega ponownej weryfikacji przy zastosowaniu testów manualnych, bądź przy pomocy narzędzi automatyzujących prace testowe (jednak innych niż stosowanych podczas fazy testów właściwych). „W świecie skanerów zabezpieczeń fałszywe trafienia są powszechnym i niefortunnym skutkiem ubocznym. Fałszywe trafienie pojawia się, gdy narzędzie testowe zgłasza obecność luki w zabezpieczeniach w miejscu, gdzie jej nie ma”[3]. Stąd bardzo ważnym elementem jest uzupełnianie w każdym kroku testowym działań automatycznych, testami manualnymi.

Tabela 6. Przykład planu testów PTER- faza III

Krok	Opis	Data od	Data do	Przewidywany czas [h]
1	Weryfikacja podatności	08.09.2015	09.09.2015	5
2	Podział i oznakowanie błędów	09.09.2015	10.09.2015	2
3	Wybrane przypadki użycia	10.09.2015	11.09.2015	1
4	Sposoby minimalizacji ryzyka	11.09.2015	12.09.2015	1
5	Stworzenie i przekazanie raportu	12.09.2015	13.09.2015	1
			Suma:	**10 [h]**

Do wszystkich błędów zostają przypisane nazwy i poziomy ryzyka stworzone przez organizację NIST. Na raporcie finalnym oprócz standardowej

listy błędów umieszczane są również błędy opisane szczegółowo (w postaci przypadków użycia wraz z odpowiednim opisem technicznym).

3.1.3. Dodatkowe wymagania

Metodyka PTER specyfikuje 2 rodzaje dokumentów realizowanych przez osobę testującą:

- notatka częściowa,
- raport końcowy.

Notatka częściowa może stanowić dowolną formę dokumentu, ponieważ jest ona realizowana jedynie do celów opisania fragmentu pracy testera, np. w celu zgłoszenia podatności, której zostanie przypisane wysokie lub krytyczne ryzyko.

Dodatkowego opisu wymaga raport, tworzony w celu podsumowania wykonanej pracy. Powinien on zawierać obowiązkowo odpowiednią klauzulę poufności, uzgodnionych adresatów oraz obszary w postaci 7 rozdziałów przedstawionych w tabeli 7.

Raport służy do przedstawienia zidentyfikowanych w ramach testów podatności oraz udokumentowania przeprowadzonych czynności testowych.

Wstęp do raportu powinien zawierać dane związane z podstawą prawną testów (np. numer umowy) oraz uzgodnionym ze stroną zlecającą i zdefiniowanym ich celem, a także przyjętymi założeniami. Założenia mogą zawierać wykluczenia związane z zakresem, funkcjami, czy miejscem ich realizacji. Dokładne zdefiniowanie tych kwestii na samym wstępie ma na celu zapobieganie ewentualnym sytuacjom spornym związanym z przeprowadzonymi pracami.

Raport powinien uwzględniać punkt, w którym krótko zostanie przedstawiona testowana aplikacja. Mogą to być podstawowe informacje uzyskane od zlecającego, związane z adresacją aplikacji, technologią użytą do jej wykonania oraz krótkim opisem jej przeznaczenia.

Raport powinien obejmować szczegółowy przebieg testów, określonych przez czynności metodyki. W przypadku braku zidentyfikowania podatności w konkretnym kroku, punkt powinien być uzupełniony o taką informację. Każdy test w wyniku którego dokonano zidentyfikowania podatności, powinien być uszczegółowiony (o ile to techniczne możliwe) zrzutem ekranu lub odnośnikiem do pliku nagrania. Podatności zgłoszone przez oprogramowanie powinny zostać zweryfikowane, czy przypadkiem nie są tzw. *false positive*. Poszczególne, potwierdzone podatności powinny posiadać przypisane numery porządkowe (sygnatury), które pozwolą w sposób jednoznaczny odwoływać się do podatności w różnych miejscach raportu.

Tabela 7. Struktura raportu końcowego PTER

	Tytuł punktu	Opis zawartości
1	Wstęp z opisem celu testów	- podstawa testów (numer umowy), - przedstawienie informacji wstępnych, - adresaci raportu, - czas trwania testów, - osoby zaangażowane, - ustalenia nt. zakresu testów, cel testów i przyjęte założenia.
2	Przedstawienie testowanej aplikacji	- uzyskane informacje podstawowe od zlecającego związane z adresacją i technologią, - krótki opis przeznaczenia aplikacji.
3	Podsumowanie dla biznesu	- wyszczególnienie zidentyfikowanych podatności, - ilość podatności o poziomach: krytycznym średnim i niskim, - krótki opis w formie nietechnicznej dotyczący podatności o najwyższym poziomie, - ilość zidentyfikowanych URL.
4	Podsumowanie techniczne zidentyfikowanych podatności	- lista podatności z podziałem na poziom ryzyka i przypisanie do rodzaju błędu, - do każdej podatności przypisane priorytety wg SANS, - do każdej podatności dodane zalecenia związane z minimalizacją ryzyka ich wykorzystania, - podział na błędy: *true positive* oraz *false positive*.
5	Szczegółowy przebieg testów	- kolejne podpunkty dotyczące każdego kroku metodyki (bez fazy III), - każdy test z potwierdzoną podatnością zawiera, o ile to technicznie możliwe, zrzut ekranu, lub sekwencję wideo, - pod zidentyfikowaną podatnością jej krótki opis.
6	Szczegółowy scenariusz możliwości wykorzystania wybranej podatności	- nazwa wybranej podatności i opis, - model przypadku (złośliwego) użycia, - scenariusz wykorzystania, - konsekwencje.
7	Załączniki	- lista załączników, - logi i raporty ze skanerów automatycznych, - własne *exploity*.

W podsumowaniu powinno znaleźć się tabelaryczne zestawienie wszystkich wykrytych i potwierdzonych podatności. Dla każdej z nich należy przedstawić krótki opis zawierający sygnaturę, kategorię podatności i odwołanie do testu, który ją ujawnił oraz przeprowadzić ogólną analizę ryzyka, przy której wzięte będzie pod uwagę prawdopodobieństwo jej wykorzystania oraz potencjalne konsekwencje wykorzystania.

Należy mieć na uwadze, że odbiorcami raportu są zarówno osoby techniczne (np. administratorzy aplikacji), jak i pracownicy z wyższego szczebla zarządzającego, nieposiadający specjalistycznej wiedzy. Dlatego też w zakresie raportu powinien się znaleźć punkt podsumowujący badania, w formie zrozumiałej dla nietechnicznego odbiorcy. Podsumowanie dla kierownictwa

powinno zawierać również pewne dane statystyczne potwierdzające np. czas poświęcony na przeprowadzenie poszczególnych faz, ilość zidentyfikowanych adresów URL, testowanych modułów aplikacji itp.

Technicznemu odbiorcy testów oprócz podsumowania zawierającego listę ze szczegółowym opisem podatności powinny zostać przedstawione sposoby minimalizacji ryzyka z nich wynikającego. Jednak nie zawsze wiedza posiadana przez wykonujących testy, np. na temat dodatkowych zabezpieczeń stosowanych do ochrony aplikacji internetowej, pozwoli na opracowanie skutecznych zaleceń tego typu.

Dla wybranej podatności z najwyższego poziomu ryzyka, należy sporządzić szczegółowy scenariusz możliwości jej wykorzystania uzupełniony o graficzny model przypadku (złośliwego) użycia. Punkt posiada również istotne znaczenie związane z realizacją założenia metodyki, dotyczącego pobudzania świadomości dotyczącej testów bezpieczeństwa u zainteresowanych stron.

Ostatnim elementem raportu jest lista załączników, którymi mogą być np. logi i raporty ze skanerów automatycznych, filmy obrazujące wykorzystanie, bądź identyfikację podatności, zastosowane w ramach testów własne kody expolit itp.

4. Podsumowanie

W artykule skupiono się na przedstawieniu założeń oraz opisie przebiegu proponowanej metodyki testowania bezpieczeństwa aplikacji internetowych-PTER. Przedstawiono również ograniczenia stworzonej metodyki oraz wprowadzono słownik ze stosowaną w ramach jej stosowania terminologią.

Metodyka PTER jest metodyką specjalistyczną, ukierunkowaną na przeprowadzanie badań związanych z bezpieczeństwem aplikacji internetowych (WWW). Położono w niej nacisk, na tzw. „ścieżkę techniczną", przeprowadzaną przy użyciu narzędzi automatyzujących pracę i uzupełnianą manualnymi testami penetracyjnymi. Istotną sprawą jest bowiem korzystanie przez osobę testującą z własnej wiedzy i doświadczenia, które mogą pozwolić na zidentyfikowanie błędów niemożliwych do wykrycia przez skanery automatyczne.

W wyniku wcześniejszego porównania dostępnych metodyk z zakresu testowania bezpieczeństwa aplikacji WWW (w szczególności OSSTM, OWASP, Agile AST, EAST) zidentyfikowano obszary wymagające pokrycia, które zostały zaadresowane w założeniach kompleksowej metodyki PTER. Dla przypomnienia oprócz własności częściowo pokrytych przez porównywane metodyki (np. zakres, liczba faz, metryki, narzędzia, środowisko testowe, raportowanie) były również obszary, niepokryte przez żadną z metodyk: oznaczanie podatności oraz repozytorium testowe [1]. Zaproponowane

repozytorium testowe wspomaga realizację założeń metodyki podczas całego procesu testowego, a w szczególności używane jest po każdym punkcie kontrolnym faz oraz wprowadza ułatwienia związane z zarządzaniem procesem wdrażania poprawek oraz oceną skuteczności podejmowanych działań.

W artykule przedstawiono obligatoryjną ścieżkę techniczną, na którą składają się 3 główne fazy zawierające kolejne kroki i odpowiadające im czynności testowe. Dla każdego kroku przedstawiono opis i jego uzasadnienie oraz oczekiwane parametry wejściowe i wyjściowe. Dodatkowo dla każdej fazy przedstawiono opis planu testów i przedstawiono przykład planu zawierającego zakresy czasowe.

Ważną cechą stworzonej metodyki PTER jest możliwość jej adaptacji do postaci uszczegółowienia dziedzinowego (tematycznego) dla metodyki szerzej traktującej tematykę, np. metodyki przeprowadzania testów penetracyjnych systemów teleinformatycznych P-PEN. Innym elementem charakteryzującym sposobność ciągłego rozwijania metodyki PTER jest możliwość ustawicznego jej uzupełniania w działaniach dostępnych w kroku IIK5. Krok ten pozwala na zachowanie w czasach szybkiego rozwoju aplikacji internetowych, odpowiedniego poziomu elastyczności stworzonej metodyki. W przyszłości punkt mógłby zawierać wyszczególnione konkretne testy dodatkowe, realizowane w zależności od potrzeb strony zlecającej, np. testy aplikacji WWW:

- dedykowanych dla urządzeń mobilnych – IIK5-x,
- dedykowanych dla urządzeń wbudowanych, czy systemów sterowania przemysłowego – IIK5-y.

gdzie x oraz y to kolejne numery porządkowe testu dodatkowego.

Kolejnym elementem zidentyfikowanym jako możliwości przyszłościowe, może być rozwinięcie przedstawionej jako opcjonalnej „ścieżki formalnej". Może się zdarzyć, że stronie zlecającej oprócz konieczności dostarczenia technicznych dowodów związanych z poziomem zaufania do testowanej aplikacji, będzie zależało również na formalnym uzasadnieniu spełnienia wymagań obowiązujących regulacji prawnych, np. wspominanej wcześniej „Ustawy o Ochronie Danych Osobowych", zgodności z normą ISO/IEC 17799 itp. Stąd ścieżka formalna powinna zostać uzupełniona w szczegółowy opis procesów tak jak w przypadku zaproponowanej „ścieżki technicznej", bazując przy tym na wspomnianych dokumentach i przyjętej tam terminologii.

Dodatkowym elementem, który można wziąć pod uwagę w rozważanych propozycjach rozwinięcia metodyki PTER, jest dostarczenie metodyki analizy ryzyka, dostosowanej do założeń metodyki testów.

Autorzy są zdania, że przeprowadzona analiza oraz praktyczne doświadczenie w przeprowadzaniu testów bezpieczeństwa aplikacji WWW dały wystarczające podstawy do stworzenia spójnej i kompleksowej metodyki,

adresującej główne problemy związane z prowadzeniem tego typu prac. Zaproponowane podejście metodyczne ma na celu zarówno uporządkowanie pracy pentesterów, jak i jasne przedstawienie zidentyfikowanych podatności potencjalnym odbiorcom (posiadającym wiedzę techniczną oraz nie posiadającym takiej wiedzy). Elastyczne podejście do czynności testowych pozwala na identyfikację wszystkich znanych podatności i błędów w testowanej aplikacji, jednocześnie pozwalając na wykorzystanie "nieopisanej" wiedzy eksperckiej.

Autorzy są przekonani, że brak metodycznego podejścia do testów bezpieczeństwa aplikacji WWW, skutkuje nierzetelnymi i z pewnością tylko częściowymi wynikami. Zgodnie z założeniami metodyki PTER zostały przeprowadzone praktyczne testy bezpieczeństwa. Dlatego zdaniem autorów przeprowadzone metodycznie testy bezpieczeństwa aplikacji WWW, zgodnie z założeniami PTER, są ważnym elementem, podnoszącym poziom zaufania oraz pozwalającym na prawidłowe odtworzenie pracy wykonanej przez pentestera (odpowiednie raportowanie wyników i dokumentowanie prac), co zapewnia powtarzalność testów umożliwiającą porównywanie wyników.

Autorzy są również zdania, że testy bezpieczeństwa nie zapewniają kompletnej obrony przed błędami bezpieczeństwa aplikacji internetowych. Są one jednak ważnym składnikiem kompleksowej ochrony, który powinien być traktowany na równi z zabezpieczeniami technicznymi takimi, jak np. systemy klasy WAF (ang. *Web Application Firewall*), IDS/IPS (ang. *Intrusion Detection System, Intrusion Prevention System*), NGF (ang. *Next Generation Firewall*). Należy dodać, że testy powinny być wykonywane cyklicznie, odpowiednio do zmian w chronionej aplikacji oraz otaczających ją zagrożeń. Z powyższych powodów istotną wydaje się kwestia wykonywania tych testów w sposób metodyczny, co z kolei implikuje potrzebę zastosowania skutecznej, kompletnej i spójnej metodyki testowania aplikacji internetowych.

Literatura

[1] ANTCZAK M., ŚWIERCZYŃSKI Z., *Metodyki testowania bezpieczeństwa aplikacji internetowych*, Przegląd Teleinformatyczny Nr 2 (20), 2003.

[2] BIAŁAS A., *Bezpieczeństwo informacji i usług w nowoczesnej instytucji i firmie*, Wydawnictwa Naukowo-Techniczne, 2007.

[3] DHANJANI N., CLARKE J. *Bezpieczeństwo sieci. narzędzia. Tworzenie i stosowanie narzędzi do testowania zabezpieczeń*, Helion, 2006.

[4] H P., WALTER B., *Testowanie bezpieczeństwa aplikacji internetowych. Receptury*, Helion, 2010.

[5] LIDERMAN K., *Podręcznik administratora bezpieczeństwa teleinformatycznego*, Mikom, 2003.

[6] PATKOWSKI A. E., *Metodyka P-PEN przeprowadzania testów penetracyjnych systemów teleinformatycznych*, „Biuletyn Instytutu Automatyki i Robotyki nr 24/2007", Wojskowa Akademia Techniczna, 2007.

[7] SILLITTI A., WANG X, MARTIN A, WHITWORTH E, *Agile Processes in Software Engineering and Extreme Programming*, 11th International Conference, XP 2010, Trondheim, Norway, June 1-4, 2010, Proceedings, str.14-26, Springer, 2010.

[8] SUBIETA K., *Wprowadzenie do inżynierii oprogramowania*, PJWSTK, 2002.

ŹRÓDŁA INNE

[9] ANTCZAK M., *Metodyka oraz dedykowany system służący do wykonywania testów bezpieczeństwa aplikacji internetowych (WWW),* Praca magisterska, Wojskowa Akademia Techniczna, 2012.

[10] Standard BS 7000-1:2008, Design management systems. Guide to managing innovation, wyd. BSI, Wielka Brytania 2008,

[11] Norma ISO 15408, Common Criteria.

[12] Polska Norma PN-ISO/IEC 11799:2007.

[13] Polska Norma PN-ISO/IEC 27001:2007.

[14] Polska Norma PN-ISO/IEC 27001:2014.

[15] Portal Wikipedia [online]. „*Red team - Wikipedia, the free encyclopedia*", [dostęp: 21-06-2015]. Dostępny w Internecie: http://en.wikipedia.org/wiki/Red_team

[16] Portal Wikipedia [online]. „*Tiger team - Wikipedia, the free encyclopedia*", [dostęp: 21-06-2015]. Dostępny w Internecie: http://en.wikipedia.org/wiki/Tiger_team

[17] Portal Wikipedia [online]. „*Webmail - Wikipedia, wolna encyklopedia*", [dostęp: 21-06-2015]. Dostępny w Internecie: http://pl.wikipedia.org/wiki/Webmail

[18] Strona Agile Methodology [online]. „*Development Methodologies | What Is Agile Methodology*", [dostęp: 21-06-2015]. Dostępny w Internecie: http://agilemethodology.org/

[19] Strona CERT.GOV.PL [online]. „*Raport z działalności zespołu CERT.GOV.PL za II kwartał 2012 roku*", [dostęp: 21-06-2015]. Dostępny w Internecie: http://www.cert.gov.pl/portal/cer/56/561/Raport_z_dzialalnosci_zespolu_CERTGOVPL_za_II_kwartal_2012.html

[20] Strona Common Weakness Enumeration [online]. „*2011 CWE/SANS Top 25 Most Dangerous Software Errors. Appendix D: Comparison to OWASP Top Ten 2010*", [dostęp: 21-06-2015]. Dostępny w Internecie: http://cwe.mitre.org/top25/#AppendixD

[21] Strona Common Weakness Enumeration [online]. „*2011 CWE/SANS Top 25 Most Dangerous Software Errors*" – wersja z 13 września 2011 roku, [dostęp: 21-06-2015]. Dostępny w Internecie: http://cwe.mitre.org/top25/

[22] Strona ISECOM [online], Pete Herzog, „*OSSTMM 3. The Open Source Security Testing Methodology manual*", [dostęp: 21-06-2015]. Dostępny w Internecie: http://www.isecom.org/mirror/OSSTMM.3.pdf

[23] Strona National Institute of Standards and Technology [offline]. „NIST 800-115 Appendix F- Glossary". [dostęp: 21-06-2015]. Dostępny w Internecie: http://csrc.nist.gov/publications/nistpubs/800-115/SP800-115.pdf

[24] Strona National Vulnerability Database (NVD) Search Vulnerabilities [online]. "CVE and CCE Vulnerability Database", [dostęp: 21-06-2015]. Dostępny w Internecie: http://web.nvd.nist.gov/view/vuln/search

[25] Strona Norwegian University of Science and Technology [online]. „*Computer and Information Science, Department of Computer and Information Science*", 2002 [dostęp: 21-06-2015]. Dostępny w Internecie: http://www.idi.ntnu.no/grupper/su/publ/ese/ieee-se-glossary-610.12-1990.pdf, str. 62

[26] Strona OWASP Foundation © 2011 [online]. „*Category: OWASP Testing Project*", [dostęp: 21-06-2015]. Dostępny w Internecie: https://www.owasp.org/index.php/OWASP_Testing_Project

[27] Strona OWASP Foundation © 2011 [online]. „*Category: OWASP Project-OWASP*", [dostęp: 21-06-2015]. Dostępny w Internecie: https://www.owasp.org/index.php/Category:OWASP_Project

[28] Strona OWASP Foundation © 2011 [online]. „Category: OWASP Top Ten Project", [dostęp: 21-06-2015]. Dostępny w Internecie: https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project

[29] Strona Scrum Methodology [online]. „*Scrum Phases overview*", [dostęp: 21-06-2015]. Dostępny w Internecie: http://www.scrummethodology.org/scrum-phases.html

[30] Schwaber K., Sutherland J., „*The Scrum Guide*", wersja polska. [dostęp: 21-06-2015]. Dostępna w Internecie: http://www.scrum.org/Scrum-Guides,

[31] Strona The Web Application Security Consortium / Threat Classification [online]. „*WASC Threat Classification online*", [dostęp: 21-06-2015]. Dostępny w Internecie: http://projects.webappsec.org/w/page/13246978/Threat%20Classification

[32] Strona University of Calgary [online], „*Electrical Engineering*". A. Tappenden, P. Beatty, J. Miller , A. Geras, M. Smith „*Agile Security Testing of Web-Based Systems via HTTPUnit*", [dostęp: 21-06-2015]. Dostępny w Internecie: http://enel.ucalgary.ca/People/Smith/2008webs/encm515_08/08ReferenceMaterial/Agile%20Security%20Testing%20Miller%20Smith.pdf

Testing of internet applications security

ABSTRACT: Article provides an the main assumptions of the methodology developed web application security testing (WWW) - PTER. The methodology focuses primarily on how to perform the tests the so-called "Technical path" and document their results. Use of "agile" approach of methodology allows to focus on the needs of the recipient of the security tests. PTER methodology allows adaptation to the form of subject-specific supplement for methodology which treats the subject of test more widely (eg. the P-PEN methodology).

KEYWORDS: internet applications security, security testing, security vulnerabilities, methodology of security testing, penetration tests, security attributes.

Praca wpłynęła do redakcji: 20.11.2014 r.