

Standardy oceny bezpieczeństwa i strategie ochrony aplikacji Web

Aplikacje Web tworzone są przez deweloperów na zamówienie firm, w celu spełnienia ich specyficznych potrzeb. Firma zatrudniając dewelopera do napisania swojej aplikacji chciałaby, aby posiadał odpowiednie kompetencje, środki i czas, pozwalające na zapewnienie bezpieczeństwa tworzonej aplikacji (tzn. aplikacja powinna być wolna od podatności).

Dowiedz się:

- o audytowaniu i projektowaniu zabezpieczeń systemów IT opartych o technologie Web
- o testowaniu aplikacji Web – standardy (m.in. OWASP ASVS) i dobre praktyki
- jak zaplanować testy bezpieczeństwa
- o architekturze zabezpieczeń aplikacji Web
- o ochronie przed specyficznymi atakami na aplikacje Web

Powinieneś wiedzieć:

- znajomość podstawowych pojęć z obszaru bezpieczeństwa IT



dr inż. Mariusz Stawowski

Zarządca Działem Usług Profesjonalnych CLICO. Posiada ponad 12-letnie doświadczenie w prowadzeniu projektów i audytów bezpieczeństwa. Otrzymał tytuł doktora nauk technicznych w Wojskowej Akademii Technicznej w Warsza-

Wtym celu deweloper powinien w cyklu rozwojowym aplikacji stosować odpowiednie metody zapewnienia bezpieczeństwa, np. Microsoft Security Development Lifecycle, OWASP Security Assurance Maturity Model, itp. W praktyce często okazuje się jednak, że deweloper aplikacji posiada ograniczone środki i czas oraz niewystarczające kompeten-

cie w obszarze bezpieczeństwa. Deweloper w pierwszej kolejności koncentruje swoje działania nie na bezpieczeństwie, ale spełnieniu wymagań funkcjonalnych, czego efektem może być oddanie do użytku aplikacji posiadającej podatności. W takiej sytuacji zapewnienie bezpieczeństwa aplikacji możliwe jest poprzez zatrudnienie kompetentnego audytora, który zidentyfikuje podatności aplikacji a deweloper poprawi aplikację na podstawie wyników audytu.

wie za pracę w dziedzinie analizy i projektowania zabezpieczeń sieciowych systemów informatycznych. Początkowo oficer bezpieczeństwa systemu informatycznego wykorzystywanego przez NATO. Jednocześnie był członkiem międzynarodowej grupy ds. bezpieczeństwa wojskowych systemów informatycznych. Członek ISSA Polska. Posiada certyfikaty CISSP, PRINCE2 Practitioner oraz stopnie specjalizacji w zakresie wiodących technologii zabezpieczeń m.in. Check Point i Juniper Networks. Autor wielu artykułów w magazynach polskich i zagranicznych oraz 6 książek o tematyce bezpieczeństwa.

Dodatkowo aplikacja może zostać wyposażona w zabezpieczenia, które utrudniają wykorzystanie posiadanych przez nią podatności, których nie udało się poprawić lub nie zostały wykryte przez audytora, tzw. „wirtualne poprawki”. Dla aplikacji Web zalecenia w tym zakresie zawarte są m.in. w standardzie PCI DSS, który mówi o potrzebie stosowania zabezpieczeń Web Application Firewall. W dalszej części opracowania zostaną dokładniej omówione zagadnienia audytowania i ochrony aplikacji Web.

Testowanie aplikacji

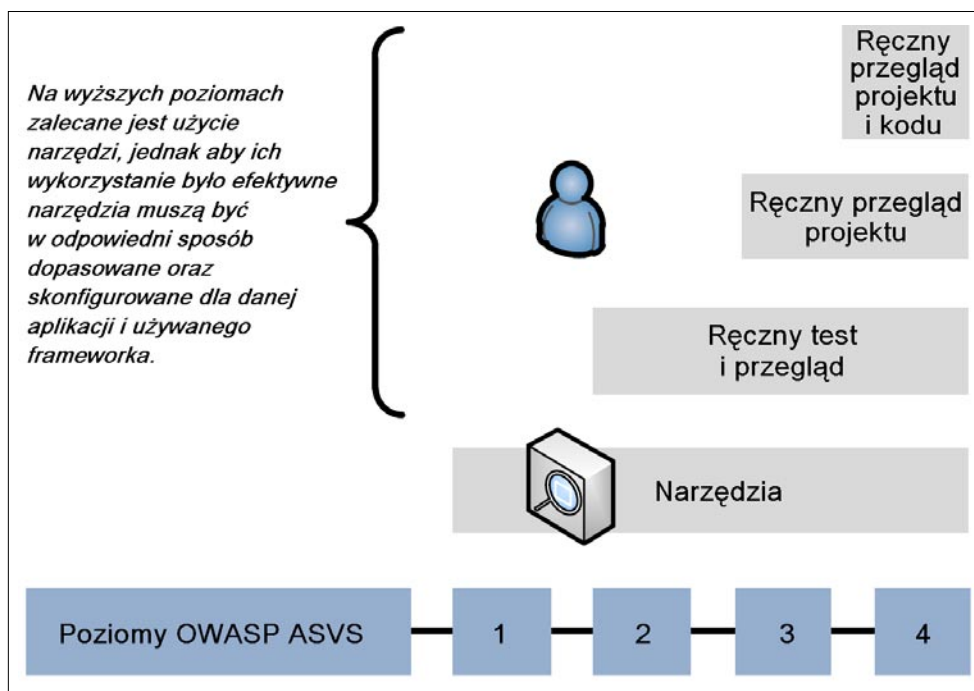
– standardy i dobre praktyki

Testy bezpieczeństwa aplikacji Web wymagają odpowiedniego podejścia, uwzględniającego ich specyficzne podatności i możliwości ich wykorzystania. Zastosowanie samych narzędzi automatycznych (tzw. skanerów Vulnerability Assessment) w przypadku aplikacji Web nie jest wystarczające i w praktyce umożliwia wykrycie tylko niewielkiej liczby błędów. Aplikacje Web tworzone na zamówienie firm w celu spełnienia ich indywidualnych wymagań biznesowych oprócz typowych podatności posiadają także unikalne podatności, szczególnie odnoszące się do błędów logiki biznesowej.

Skuteczny audyt aplikacji Web może zostać wykonany w oparciu o dokumenty i zalecenia organizacji Open Web Application Security Project (OWASP). Organizacja OWASP publicznie udostępnia wiele dokumentów przydatnych dla audytorów. Próbą standaryzacji zakresu testowania aplikacji Web jest dokument Application Security Verification Standard (ASVS). Polska wersja dokumentu OWASP ASVS dostępna jest pod adresem:

<http://owasp-asvs.googlecode.com/files/asvs-webapp-release-2009-pl.pdf>

ASVS to dokument opisujący wymagania względem weryfikacji projektu, implementacji i właściwego użycia mechanizmów bezpieczeństwa aplikacji Web. Zawiera cztery poziomy weryfikacji jej bezpieczeństwa. Każdy z tych poziomów definiuje określone wymagania, które audytor może wykorzystać w trakcie testów efektywności działania mechanizmów bezpieczeństwa chroniących aplikacje Web. Obszar weryfikacji odnosi się do bezpieczeństwa komponentów samej aplikacji (bez elementów systemu operacyjnego i infrastruktury sieciowej), a także bezpieczeństwa jej danych pod-



Rysunek 1. Poziomy weryfikacji bezpieczeństwa aplikacji Web (źródło: OWASP ASVS)

czas transmisji pomiędzy komponentami aplikacji, pomiędzy klientami i serwerami oraz pomiędzy systemami zewnętrznymi a aplikacją. Dla poszczególnych poziomów ASVS ustalone są także odpowiednie wymagania raportowe.

ASVS definiuje wymagania dla mechanizmów bezpieczeństwa aplikacji Web. Nie opisuje w jaki sposób audytor może wykonać testy i jakich podatności powinien wyszukiwać. Rekomendowane narzędzia i techniki testowania aplikacji Web oraz podatności, pod kątem których należy wykonać testy można znaleźć w innych dokumentach, m.in. OWASP Testing Guide, OWASP Code Review Guide, OWASP Top 10. Rysunek 1 prezentuje poziomy weryfikacji bezpieczeństwa aplikacji Web - od poziomu 1 do 4. Obowiązuje zasada, że przechodząc na wyższy poziom rośnie szerokość i głębokość weryfikacji. Szerokość weryfikacji oznacza wymagania jakie muszą być spełnione - każdy z wyższych poziomów ASVS dodaje kolejne wymagania dla mechanizmów bezpieczeństwa aplikacji. Głębokość weryfikacji oznacza podejście i rygorystyczność weryfikacji.

Tabela 1. Wybór poziomu ASVS ze względu na zagrożenia aplikacji

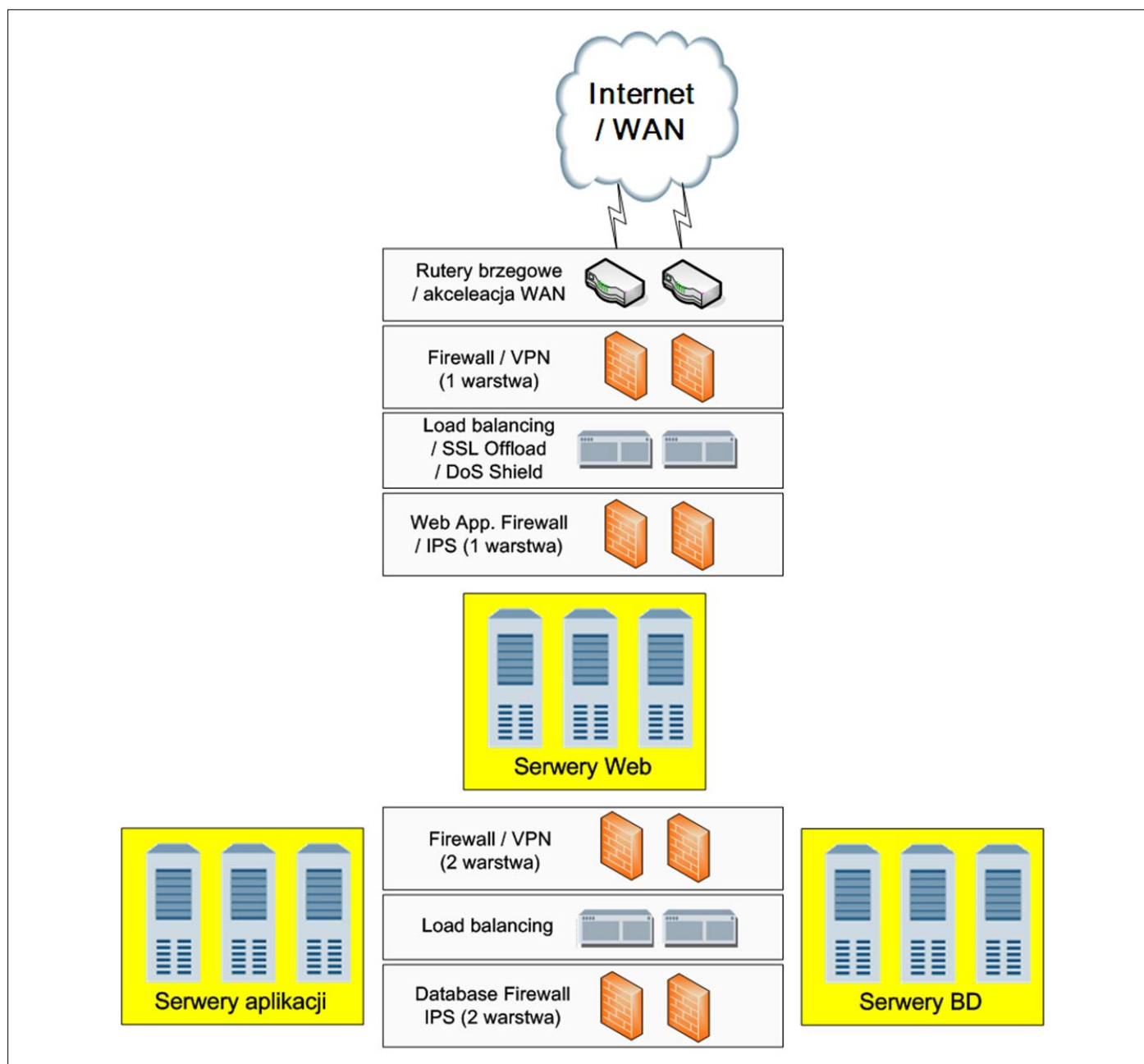
| Poziom ASVS | Zagrożenia aplikacji |
|-------------|---|
| 1 | Złośliwe aplikacje jak np. wirusy i robaki, gdzie cele są wybierane przypadkowo w trakcie skanowania o szerokim zakresie i atakowane są aplikacje najbardziej podatne. |
| 2 | Złośliwe aplikacje jak np. wirusy i robaki oraz mało wymyślni (mało doświadczeni, wykształceni), liczący na okazję napastnicy, używający profesjonalnych narzędzi ataku lub narzędzi open-source. |
| 3 | Złośliwe aplikacje jak np. wirusy i robaki, napastnicy liczący na okazję oraz zdeterminowani napastnicy (wykwalifikowani i zmotywowani, skupieni na określonych celach, używający m.in. specjalnie przystosowanych narzędzi skanujących). |
| 4 | Zdeterminowani napastnicy - wykwalifikowani i zmotywowani, skupieni na określonych celach, używający m.in. specjalnie przystosowanych narzędzi skanujących. |

ASVS zakłada, że audytor przyjmując zakres testów aplikacji na poziomie I będzie wykrywał podatności za pomocą automatycznych narzędzi (np. skanerów wykorzystujących wzorce podatności). Poziom I dzieli się na IA i IB. Poziom IA to weryfikacja dynamiczna, gdzie audytor skanuje aplikację podczas jej pracy. Poziom IB to weryfikacja statyczna, gdzie audytor za pomocą automatycznych narzędzi poddaje analizie kod źródłowy aplikacji. ASVS zakłada przy tym, że wszystkie podatności wykryte na poziomie I za pomocą narzędzi automatycznych powinny zostać ręcznie potwierdzone przez audytora. Narzędzia automatyczne mogą bowiem dostarczyć błędnych lub niepełnych wyników.

Poziom 2 to weryfikacja ręczna, gdzie audytor także może wykorzystywać różne narzędzia (np. proxy prze-

chwytujące, fuzzery, edytory kodu oprogramowania), jednak nie powinny to być tylko skanery (przewidziane dla poziomu I), które pełnią tu rolę pomocniczą. Poziom 2 dzieli się na 2A, gdzie wykonywane są ręczne testy penetracyjne aplikacji oraz 2B, w którym audytor wykonuje ręczną weryfikację kodu źródłowego. Poziom 2 zapewnia znacznie wyższą wykrywalność podatności od poziomu I. W praktyce bowiem za pomocą narzędzi automatycznych nie ma możliwości wykrycia wielu poważnych podatności aplikacji Web (np. błędów logiki biznesowej, błędów projektowych, podatności w procesach wieloetapowych i wielu innych podatności, których wykrycie jest niemożliwe przy wykorzystaniu skanerów).

Wyższe poziomy ASVS wymagają od audytora wykonania testów o większej szerokości i głębokości. Po-



Rysunek 2. Typowa architektura aplikacji opartej na technologii Web (e-commerce, e-banking, itp.)

ziom 3 zakłada wykonanie dokładnej, ręcznej weryfikacji bezpieczeństwa aplikacji (m.in. oceny zastosowanych w aplikacji modułów kryptograficznych), a także udokumentowania architektury bezpieczeństwa aplikacji oraz wykonania modelowania zagrożeń i na ich podstawie oceny poprawności projektu i użycia wszystkich mechanizmów bezpieczeństwa. ASVS rekomenduje zastosowanie poziomu 3 dla aplikacji obsługujących ważne transakcje biznesowe pomiędzy organizacjami, włączając w to aplikacje przetwarzające dane o stanie zdrowia, aplikacje spełniające krytyczne dla biznesu lub wrażliwe funkcje lub przetwarzające inne wrażliwe aktywa.

Dla poziomu 4 standard ASVS zakłada wykonanie jeszcze bardziej dokładnie ręcznej weryfikacji bezpieczeństwa aplikacji, a w tym analizy całości kodu napisanego lub zmodyfikowanego w aplikacji pod kątem wyszukiwania złośliwego kodu. Złośliwy kod rozumiany jest w tym przypadku jako kod włączony do aplikacji podczas jej tworzenia, bez wiedzy właściciela aplikacji, który omija przyjętą politykę zabezpieczeń aplikacji. Nie jest to złośliwy kod w rozumieniu malware (np. wirus, robak). Poziom 4 jest zwykle odpowiedni dla krytycznych aplikacji, od których zależy życie i bezpieczeństwo, krytyczna infrastruktura lub zadania związane z obronnością. Poziom ten zapewnia, że mechanizmy bezpieczeństwa pracują prawidłowo, są użyte w aplikacji wszędzie tam, gdzie są potrzebne do egzekwowania polityk spe-

cyficznych dla aplikacji oraz przestrzegania praktyk pisania bezpiecznego kodu.

Zakres ASVS obejmuje komponenty aplikacji Web. ASVS nie definiuje wymagań dla weryfikacji bezpieczeństwa platformy aplikacji Web (m.in. systemu operacyjnego, serwera aplikacyjnego, serwera bazy danych, wirtualnej maszyny), ani też infrastruktury sieci i zabezpieczeń (ruterów, przełączników, urządzeń zabezpieczeń, systemów zarządzania). Wytyczne do wykonania testów infrastruktury aplikacji Web można znaleźć w innych dokumentach, m.in.:

- ISO/IEC 27001:2005 – do weryfikacji systemu zarządzania bezpieczeństwem,
- Payment Card Industry Data Security Standard (PCI DSS) – do weryfikacji bezpieczeństwa danych kartowych,
- Open Source Security Testing Methodology (OSSTM) – do weryfikacji zabezpieczeń infrastruktury sieciowej i systemowej.

Standard PCI DSS został opracowany z myślą o zapewnieniu bezpieczeństwa systemów informatycznych, gdzie przetwarzane są dane kartowe (np. numery kart kredytowych). Standard ten przedstawia konkretne wymagania dotyczące środków bezpieczeństwa jakie powinny zostać zastosowane, m.in. sieciowy firewall, Intrusion Prevention System oraz Web Application Firewall.

Tabela 2. Zakres i sposób weryfikacji bezpieczeństwa na poszczególnych poziomach ASVS

| Poziom ASVS | Zakres weryfikacji aplikacji | Sposób weryfikacji mechanizmów bezpieczeństwa |
|-------------|---|--|
| 1 | Całość kodu napisanego lub zmodyfikowanego w celu stworzenia aplikacji. | Dla poziomu 1A dynamiczne skanowanie aplikacji (tzn. automatyczne narzędzia skanują aplikację w trakcie jej działania pod kątem wyszukiwania błędów w oparciu o wzorce znanych podatności). Dla poziomu 1B skanowanie kodu źródłowego aplikacji (tzn. automatyczne narzędzia skanują kod źródłowy aplikacji pod kątem wyszukiwania błędów w oparciu o wzorce znanych podatności). Uwaga: Wyniki skanowania (wykryte podatności) są weryfikowane poprzez ręczne testy penetracyjne aplikacji lub przegląd kodu. |
| 2 | Jak dla poziomu 1, a także kod wszystkich pochodzących z firm trzecich frameworków, bibliotek i funkcjonalności bezpieczeństwa usług, realizujących lub wspomagających bezpieczeństwo aplikacji. | Dla poziomu 2A ręczny test penetracyjny aplikacji mający na celu ocenę poprawności projektu, implementacji i użycia mechanizmów bezpieczeństwa. Dla poziomu 2B ręczny przegląd kodu źródłowego aplikacji. Ręczny przegląd kodu polega na przeszukiwaniu i analizie (wykonywane przez człowieka) kodu źródłowego aplikacji w celu weryfikacji projektu, implementacji i właściwego użycia mechanizmów bezpieczeństwa. Oczekuje się, że analiza tego typu jest wspomagana narzędziami, jednak audytor może wykorzystywać powszechnie dostępne narzędzia, np. edytor kodu źródłowego. |
| 3 | Jak dla poziomów 1 i 2, a także kod wszystkich pochodzących z firm trzecich frameworków, bibliotek i usług związanych z aplikacją. | Szczegółowa weryfikacja bezpieczeństwa aplikacji oraz dokumentacja architektury bezpieczeństwa, a także weryfikacja poprawności projektu i użycia wszystkich mechanizmów zabezpieczeń poprzez wykonanie modelowania zagrożeń. |
| 4 | Jak dla poziomów 1, 2 i 3, a także całość pozostałego kodu związanego z aplikacją, włączając w to frameworki, biblioteki, środowiska uruchomieniowe oraz narzędzia służące do rozwoju, budowania i wdrażania aplikacji. | Szczegółowa weryfikacja bezpieczeństwa aplikacji (jak na poziomie 3) oraz dokładna analiza całości kodu napisanego lub zmodyfikowanego w aplikacji pod kątem wyszukiwania złośliwego kodu. |

Zawarte w PCI DSS wymagania uwzględniają rzeczywiste zagrożenia i mogą być zastosowane dla wszystkich aplikacji o podwyższonych wymaganiach bezpieczeństwa (np. e-commerce, e-banking, portale biznesowe), bez względu na to czy przetwarzane są tam dane kartowe. Organizacja PCI Security Standards Council publicznie udostępnia dokumenty, które mogą zostać użyte do tego celu, m.in. PCI DSS Requirements and Security Assessment Procedures.

Jak zaplanować testy bezpieczeństwa?

Planując wykonanie testów bezpieczeństwa aplikacji Web należy ustalić zakres prac audytowych oraz na jakim etapie cyklu rozwojowego aplikacji powinny one zostać wykonane. Nie jest to zadanie łatwe. Wykonanie bardziej dokładnych testów (wyższy poziom ASVS) zwykle oznacza dłuższy czas realizacji, większe zaangażowanie firmy w prace audytowe i wyższe koszty. W trakcie wyboru odpowiedniego poziomu ASVS do weryfikacji bezpieczeństwa aplikacji Web warto zastanowić się na jakie zagrożenia narażona jest ta aplikacja. Można przy tym skorzystać z tabeli 1, gdzie przedstawione są wskazówki zawarte w dokumencie ASVS. W miarę możliwości firma powin-

na wykonać analizę ryzyka i na tej podstawie ustalić wymagania bezpieczeństwa dla aplikacji, w tym również wymagania audytowe.

Należy także pamiętać o zapewnieniu odpowiednich, stabilnych warunków i środowiska testowania. Zalecane jest aby w trakcie testów aplikacja nie była modyfikowana przez dewelopera (tzn. testy funkcjonalne oraz wynikające z nich zmiany w kodzie aplikacji i testy bezpieczeństwa nie powinny być wykonywane w tym samym czasie). Zmiany w kodzie aplikacji wprowadzane przez dewelopera w trakcie trwania testów bezpieczeństwa mogą sprawić, że podatności powstałe w wyniku tych zmian nie zostaną przez audytora wykryte.

W celu właściwego wyboru poziomu ASVS, oszacowania czasochłonności prac audytorskich i wynikających z tego kosztów można także posłużyć się tabelą 2. Przedstawione są tam zakresy oraz podejścia do weryfikacji mechanizmów bezpieczeństwa aplikacji Web dla poszczególnych poziomów ASVS. W sytuacji gdy, testy muszą zakończyć się w określonym, relatywnie krótkim czasie (np. ze względu na potrzebę oddania aplikacji do użytku w krótkim terminie) warto jest także zdefiniować jakie podatności będą wyszukiwane.

The screenshot displays a WAF configuration interface. On the left, a tree view titled 'Struktura chronionej aplikacji Web' shows a hierarchy of URLs and parameters. The tree includes paths like /active_auctions.php, /browse.php, /closed_auctions.php, /contactus.php, and /edit_data.php, with various parameters such as user_id, id, TPL_address, TPL_birthdate, TPL_city, TPL_country, TPL_email, TPL_nletter, TPL_password, TPL_phone, TPL_prov, TPL_repeat_password, TPL_zip, and action. In the center, a section titled 'Dozwolone URL i parametry' shows the selected parameter configuration. On the right, a section titled 'Dozwolone wartości parametrów (kontrola typu, rozmiaru, itd.)' shows the 'Edit Parameter' settings for the parameter '[Explicit] id'. The settings include: Parameter Name: [Explicit] id; Parameter Level: URL Parameter; URL Path: HTTP /feedback.php; Perform Staging: unchecked; Allow Empty Value: checked; Allow Repeated Occurrences: unchecked; Sensitive Parameter: unchecked; Parameter Value Type: User-input value; Data Type: Integer; Check Minimum Value: 0; Check Maximum Value: 100000; Check Maximum Length: 6; Regular Expression: unchecked.

Rysunek 3. Profil ochrony aplikacji Web zbudowany przez zabezpieczenia WAF

Można posłużyć się tutaj dokumentem OWASP Top 10. Dla przykładu, zespół audytowy CLICO w trakcie ustalania zakresu prac zwykle przedstawia następujące opcje względem głębokości i rygorystyczności testów:

- Opcja 1. Testy za pomocą narzędzi automatycznych (m.in. skanery, fuzzery) z ręczną weryfikacją wyników – zakres poziomu IA standardu OWASP ASVS.
- Opcja 2. Testy ręczne wspomagane za pomocą narzędzi automatycznych (zakres poziomów IA i 2A OWASP ASVS, bez IB) w zakresie identyfikacji najbardziej krytycznych podatności (tzn. OWASP Top 10 oraz błędy logiki biznesowej) zgodnie z rekomendacjami OWASP Testing Guide.
- Opcja 3. Kompletnie testy bezpieczeństwa aplikacji zgodnie z rekomendacjami OWASP Testing Guide o głębokości i rygorystyczności wskazanego poziomu OWASP ASVS oraz inne testy wymagane przez zlecającą.

Istotne pytanie to na jakim etapie cyklu rozwojowego aplikacji Web należy zatrudnić audytora - czy już na etapie definiowania wymagań, projektowania, implementacji, czy tylko przed samym wdrożeniem aplikacji do eksploatacji? Czy po dokonaniu modyfikacji aplikacji testy powinny zostać wykonane na nowo? Obowiązuje tutaj zasada, że im wcześniej wykryte zostaną podatności tym mniejsze będą koszty ich naprawienia. Dla przykładu, usunięcie błędu projektu wykrytego w czasie testów przed-wdrożeniowych może być bardzo kosztowne i spowodować powstanie nowych błędów. W praktyce zamiast usuwania takich błędów szuka się rozwiązania zastępczego, które w pewnym zakresie zredukuje zagrożenie. Z drugiej jednak strony koszt zabezpieczeń, w tym audytów powinien być dla firmy uzasadniony ekonomicznie - powinna być zachowana adekwatność kosztów bezpieczeństwa do wartości chronionych zasobów i potencjalnych strat jakie firma może ponieść w razie naruszenia bezpieczeństwa (np. utrata zaufania klientów i reputacji).

Testy aplikacji na urządzenia mobilne

Planując wykonanie testów bezpieczeństwa istotne jest, aby zwrócić uwagę, czy aplikacja lub jej część przeznaczona jest na zwykle komputery PC czy na urządzenia mobilne (jak np. smartfon, tablet i PDA). Z jednej strony incydentów bezpieczeństwa na urządzeniach mobilnych jest jeszcze bardzo niewiele w porównaniu do liczby incydentów na komputerach PC. Wielu analityków uważa, że urządzenia mobilne są dla użytkowników znacznie bezpieczniejsze od komputerów PC. Przede wszystkim trudniej jest stworzyć złośliwy kod, który będzie skutecznie atakował dużą liczbę użytkowników urządzeń mobilnych. Na komputerach PC wystarczy przygotować taki kod dla jednego systemu operacyjnego. Z systemu Mi-

crosoft Windows korzysta bowiem większość użytkowników komputerów PC na świecie. Urządzenia mobilne wykorzystują wiele różnych platform i systemów operacyjnych, m.in. Apple iOS, Google Android OS, Nokia Symbian OS, Microsoft Windows Mobile/Phone, BlackBerry OS.

Z technicznego punktu widzenia także trudniej jest wyszukać błędy aplikacji działających na urządzeniach mobilnych niż komputerach PC. Analiza bezpieczeństwa aplikacji zainstalowanej na urządzeniu mobilnym wymaga zwykle przygotowania odpowiedniego środowiska (m.in. symulatora smartfona na komputerze PC). Dla każdego systemu operacyjnego smartfona trzeba zbudować inne środowisko testowe. Dotychczas większość incydentów związanych z smartfonami polegała na rozpowszechnianiu złośliwych aplikacji typu Trojan, umożliwiających intruzom kontrolę urządzeń mobilnych. Aplikacje te użytkownicy musieli sami zainstalować na swoich urządzeniach. I tu także jest trudniej, ponieważ producenci urządzeń mobilnych starają się, aby aplikacje przeznaczone na ich systemy były instalowane z centralnych, odpowiednio kontrolowanych repozytoriów (m.in. Apple App Store, Android Market, Windows Marketplace). Dystrybucja aplikacji przeznaczonych na komputery PC nie jest praktycznie w żaden sposób kontrolowana i ogromna liczba złośliwych aplikacji dostaje się taką metodą na komputery (wg raportu „Analiza incydentów naruszających bezpieczeństwo teleinformatyczne” wydanego przez CERT Polska za rok 2010 popularne w naszym kraju serwisy np. wrzuta.pl i przeklej.pl także służyły do dystrybucji złośliwego oprogramowania).

Z drugiej strony urządzenia mobilne są w porównaniu do komputerów PC znacznie bardziej narażone na dostęp osób nieupoważnionych, w tym kradzież i zagubienie. Testy bezpieczeństwa aplikacji na urządzenia mobilne powinny zawierać bardzo dokładną analizę jakie dane aplikacji są na urządzeniach mobilnych zapisywane, m.in. w pamięci cache, w plikach tymczasowych, w logach, czy w lokalnej bazie danych (np. SQLite w urządzeniach Apple iPhone/iPad). Audytor powinien zwrócić przy tym uwagę, jakie poufne informacje aplikacji są wyświetlane na ekranie urządzeń mobilnych i czy są odpowiednio maskowane. Znane są przypadki zapamiętywania rzutów ekranu przez złośliwe aplikacje, a także same serwisy systemu operacyjnego urządzenia mobilnego (np. w urządzeniach iPhone).

Planując testy aplikacji na urządzenia mobilne trzeba także uwzględnić, czy dostęp do aplikacji odbywa się przez przeglądarkę Web, czy oprogramowanie klienta instalowane na urządzeniach (tzw. native application). W pierwszym przypadku testy bezpieczeństwa można przeprowadzić na analogicznej zasadzie, jak dla zwykłej aplikacji Web (np. zgodnie z OWASP Testing Guide). Jeżeli jednak aplikacja jest instalowana na urządzeniach audytor powinien zadbać o przygotowanie odpowiedniego środowiska

testowania (np. narzędzi monitorujących pliki jakie aplikacja klienta zapisuje na urządzeniu mobilnym).

Standardy weryfikacji bezpieczeństwa dla urządzeń mobilnych są dopiero w trakcie tworzenia (np. OWASP Mobile Security Project). Urządzenie mobilne może być jednak przez audytora potraktowane analogicznie jak komputer PC. Testowanie bezpieczeństwa aplikacji, której klient działa na urządzeniu mobilnym może zostać wykonane na takich samych zasadach jak dla aplikacji klienta instalowanej na komputerach PC. Z technicznego punktu widzenia urządzenie mobilne to "mały komputer". Audytor powinien zweryfikować jakie protokoły komunikacji są stosowane przez aplikację na urządzeniu (np. czy klient używa HTTP/HTTPS do komunikacji z usługami typu Webservice, czy też korzysta w innego, własnego protokołu) i na tej podstawie wybrać i przygotować odpowiednie narzędzia i metody testowania.

Jak wybrać wykonawcę testów?

ASVS posiada szeroki zakres zastosowań. Może być przydatny dla audytorów wykonujących testy penetracyjne aplikacji Web, jak również analizę jej kodu źródłowego. Opisane w standardzie ASVS punkty weryfikacyjne pozwalają na wykonanie oceny mechanizmów bezpieczeństwa aplikacji niezależnie od tego, czy audytor będzie wykonywał testy dynamiczne (np. skanowanie działającej aplikacji), czy statyczne (np. skanowanie kodu źródłowego). Dopuszczalne jest także, aby zakres testów ustalony na pewnym poziomie ASVS został rozszerzony o określone punkty weryfikacyjne wyższych poziomów.

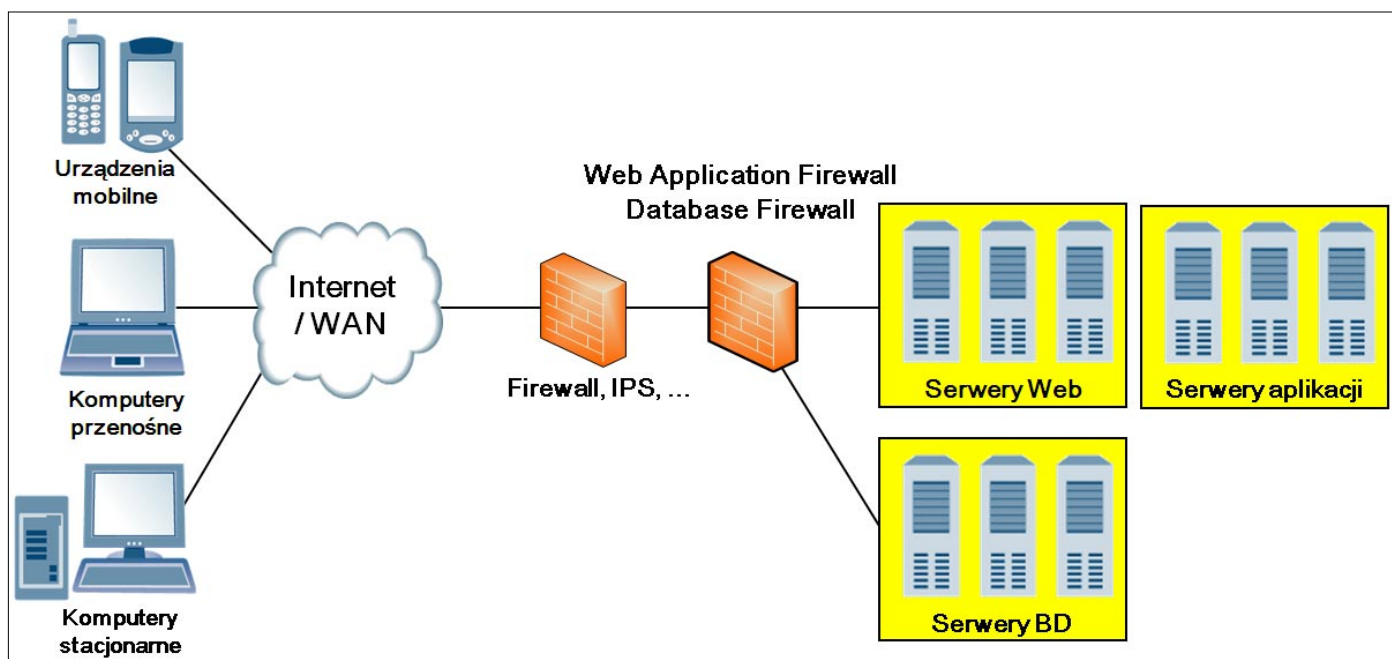
Standard może także okazać się cenny dla firm, zlecających wykonanie testów aplikacji Web do precyzyjnego zdefiniowania zakresu testów oraz oczekiwań względem

wyników prac audytora. ASVS oprócz wymagań względem mechanizmów bezpieczeństwa opisuje także wymagania raportowe. Również deweloperzy mogą użyć ASVS jako wytycznych do tworzenia aplikacji Web wyposażonych w odpowiednie zabezpieczenia. W trakcie ustalenia wymagań dla nowych aplikacji firmy mogą użyć ASVS, aby zdefiniować wymagane w aplikacji mechanizmy bezpieczeństwa.

Wymagania weryfikacyjne ASVS odnoszą się do następujących obszarów:

- Architektura bezpieczeństwa.
- Uwierzytelnianie.
- Zarządzanie sesją.
- Kontrola dostępu.
- Walidacja wejścia.
- Enkodowanie/escapowanie wyjścia.
- Kryptografia.
- Obsługa błędów i logowanie.
- Ochrona danych.
- Bezpieczeństwo komunikacji.
- Bezpieczeństwo HTTP.
- Konfiguracja zabezpieczeń.
- Wyszukiwanie złośliwego kodu.
- Bezpieczeństwo wewnętrzne.

W powyższych obszarach zawierają się określone podatności aplikacji Web. Dla przykładu w obszarze walidacji wejścia audytor wyszukuje podatności typu Cross-Site Scripting, SQL-Injection, itp. ASVS nie określa dokładnej listy podatności jakich należy wyszukiwać. Technologie stosowane przez aplikacje Web są dynamicznie rozwijane a wraz z nimi pojawiają się nowe podatności.



Rysunek 4. WAF jako dedykowana warstwa ochrony aplikacji Web

W tym przypadku głównie od kompetencji audytora zależy czy zostanie to wykonane właściwie. Samo wykonanie testów zgodnie ze standardem ASVS nie oznacza jeszcze skutecznie wykonanych testów aplikacji i znalezienia jej podatności. Kompetencje audytora oraz czas jakim dysponuje są tu bardzo ważne.

Jedną z najgroźniejszych w skutkach podatności są błędy logiki biznesowej aplikacji. ASVS nie określa na jakim poziomie należy wykonać w tym zakresie analizę. Przykładów błędów logiki biznesowej może być bardzo wiele, np.: zakup produktów za niższą cenę niż jest to ustalone przez sprzedawcę, uzyskanie kredytu z niższym oprocentowaniem niż jest to ustalone przez bank, uzyskanie kwoty kredytu przekraczającej przyznany klientowi limit, założenie lokaty w kwocie wyższej od posiadanych środków na koncie, zamówienie w sklepie internetowym produktów za ujemną kwotę i w konsekwencji zwrot gotówki, wykonanie w systemie e-banking ujemnego przelewu na inne konto i w konsekwencji pobranie środków z tego konta, itd. Podatności w obszarze logiki biznesowej mogą wynikać z błędów różnych mechanizmów bezpieczeństwa aplikacji, m.in. kontroli dostępu i walidacji wejścia. Ich identyfikacja wymaga od audytora dużej wiedzy, doświadczenia i czasu na wykonanie analizy.

Jak ocenić kompetencje audytora? W tym zakresie pomocne mogą być pisemne referencje wydane przez klientów (wraz z oświadczeniem firmy audytorskiej, że referencje te zostały wydane za pracę osób, które w dalszym ciągu są zatrudnione), a także kompetencje członków zespołu poparte wieloletnim praktycznym doświadczeniem w testowaniu aplikacji Web oraz certyfikatami (np. CISSP, specjalizacje inżynierów zabezpieczeń). Audytor może także praktycznie zaprezentować swoje kompetencje. Dla przykładu, zespół audytorski CLICO organizuje dla kadry IT odpowiedzialnej za utrzymanie bezpieczeństwa testowanych systemów warsztaty techniczne przygotowujące do współpracy z audytorem. W trakcie warsztatów omawiane są etapy prac, stosowane metodyki i narzędzia, zasady komunikacji, format i zawartość raportów, itp. W audytach na większą skalę zalecane jest zastosowanie metodyki zarządzania projektami (np. PRINCE2, PMI), gdzie już na etapie inicjowania projektu firma może ocenić kompetencje audytora oraz dokładnie ustalić produkty końcowe audytu.

Architektura zabezpieczeń aplikacji Web

Ochrona aplikacji Web wymaga zastosowania środków bezpieczeństwa adekwatnych do rzeczywistych zagrożeń oraz regularnego audytowania portalu (czego dotyczyła pierwsza część opracowania) wraz z rozwojem oprogramowania. Są to zagadnienia złożone projektowo i technologicznie. Podstawą bezpieczeństwa jest wybór właściwej architektury bezpieczeństwa i strategii ochrony. Na rysunku 2 przedstawiona została typowa architektura systemu tej klasy.

Projektanci zabezpieczeń systemów opartych na technologii Web muszą zmierzyć się z zagadnieniami specyficznymi dla tej klasy aplikacji:

- ochrona przed włamaniami wykorzystującymi błędy kodu aplikacji Web (m.in. SQL-Injection, błędy logiki biznesowej) i ataki w ruchu szyfrowanym HTTPS (SSL),
- utrzymanie dostępności usług, a w szczególności ochrona przed atakami DoS skierowanymi na aplikację Web (m.in. zalewanie aplikacji zapytaniami HTTP GET/POST).

Ochrona przed specyficznymi atakami na aplikację Web

Zabezpieczenia sieciowe firewall i Intrusion Prevention System (IPS) są podstawą do tworzenia architektury bezpieczeństwa sieci (m.in. stref bezpieczeństwa) oraz przeciwdziałania wielu groźnym atakom sieciowym (m.in. exploity na serwisy sieciowe i system operacyjny, propagacja robaków sieciowych, itd.). Zapewnienie bezpieczeństwa aplikacji Web z wykorzystaniem samych konwencjonalnych zabezpieczeń jest w praktyce niemożliwe. Stosują one bowiem techniki ochrony oparte o sygnatury (wzorce ataków). Aplikacje Web zwykle pisane są na zamówienie i posiadają unikalne podatności, których twórcy zabezpieczeń IPS nie znają i nie potrafią utworzyć dla nich odpowiednich sygnatur.

Skuteczną ochronę dla aplikacji Web zapewnia nowa kategoria zabezpieczeń Web Application Firewall (WAF). System zabezpieczeń WAF bazuje w głównej mierze na automatycznie tworzonym i aktualizowanym profilu chronionej aplikacji Web. WAF "uczy się" struktury aplikacji, URL, parametrów, cookie, itp. Tworzenie profilu ma na celu niezależne odwzorowanie oczekiwanych, poprawnych zachowań użytkowników przy dostępie do aplikacji Web. Rysunek 3 pokazuje fragment profilu aplikacji Web utrzymywanej przez zabezpieczenia WAF.

Teoretycznie większość ataków Web mogłaby zostać wykryta np. przez IPS po napisaniu odpowiednich sygnatur dla tych konkretnych ataków. Do napisania sygnatur IPS wymagane są jednak informacje, jak ataki zostaną wykonane. W rzeczywistości jest to niemożliwe, ponieważ producenci IPS nie wiedzą jakie dokładnie błędy mają aplikacje Web napisane dla firm i w jaki sposób te błędy mogą zostać wykorzystane. Ataki XSS, SQL-Injection, itd. mogą zostać wykonane na wiele różnych sposobów. Nie ma możliwości aby napisać sygnatury IPS dla wszystkich tych ataków. Dlatego też IPS za pomocą tzw. uniwersalnych sygnatur wykrywa tylko podstawowe ataki Web wykonywane w typowy sposób. Dla IPS sygnatury są podstawowym mechanizmem detekcji. WAF także wykorzystuje sygnatury ataków, ale jest to technika uzupełniająca.

Zasadnicza różnica pomiędzy zabezpieczeniami IPS i WAF polega na podejściu do kontroli ruchu sieciowego.

IPS stosuje selekcję negatywną (tzw. blacklist approach), polegającą na tym, że przepuszcza cały ruch za wyjątkiem pakietów, które zostały zidentyfikowane jako niedozwolone. W konsekwencji wszystkie ataki nierozpoznane przez IPS dochodzą do aplikacji Web. WAF stosuje selekcję pozytywną (tzw. whitelist approach), polegającą na tym, że w oparciu o zbudowany profil chronionej aplikacji Web przepuszcza wyłącznie ruch, który został zidentyfikowany jako dozwolony. Dzięki temu ruch nierozpoznany jako dozwolony jest przez WAF blokowany i ataki nie dochodzą do aplikacji Web. Więcej informacji na temat skuteczności różnych zabezpieczeń aplikacji Web można znaleźć w artykule dostępnym na stronie:

<http://www.clico.pl/edukacja/artykuly/web-ochrona>

Różne strategie ochrony aplikacji Web

Wybór odpowiedniej strategii ochrony aplikacji Web jest uzależniony od specyficznych wymagań projektu (m.in. struktury aplikacji) oraz posiadanych już urządzeń sieci i zabezpieczeń. Występują dwie główne strategie ochrony aplikacji Web, obie oparte o firewalle aplikacyjne Web:

1. WAF wdrożony jako dedykowana warstwa ochrony aplikacji Web.

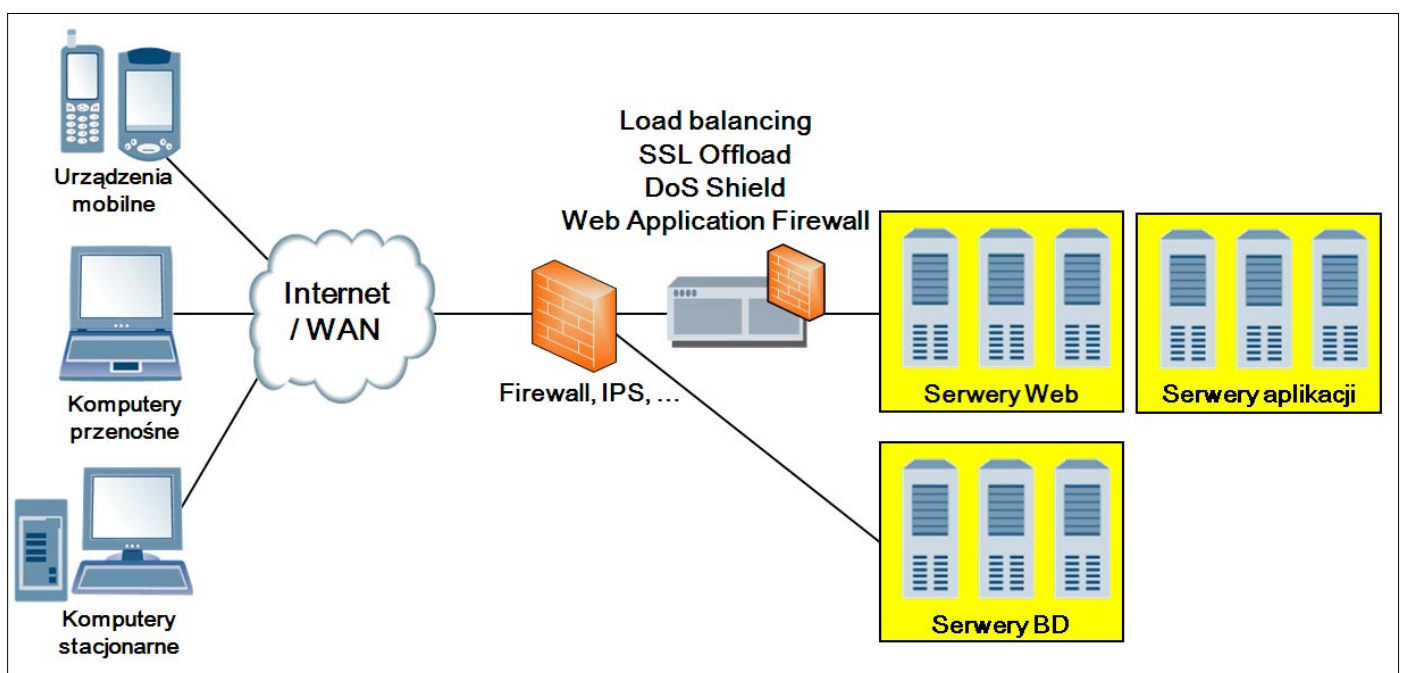
W wielu portalach biznesowych i innych aplikacjach Web (np. e-commerce, e-banking) razem z aplikacją Web ochrony wymaga także baza danych, z której korzysta aplikacja. Rozwiązanie zabezpieczeń w jednej platformie dostarcza wtedy dedykowane zabezpieczenia WAF oraz Database Firewall (DBF). Całość zabezpieczeń aplikacji Web i baz danych zarządzana jest z przeznaczone-



go tylko do tego celu systemu zarządzania, wyposażonego w odpowiednie narzędzia monitorowania i raportowania. Koncepcja wdrożenia zabezpieczeń WAF w takiej architekturze została przedstawiona na rysunku 4.

2. WAF wdrożony jako zintegrowany element infrastruktury sieciowej.

Moduł WAF może działać na zintegrowanej platformie sprzętowej, zapewniającej także zwiększenie wydajności, optymalizację i ochronę aplikacji przed awariami (load balancing), sprzętową obsługę operacji kryptograficznych (SSL offload), akcelerację Web oraz ochronę przed sieciowymi i aplikacyjnymi atakami D/DoS. Koncepcja wdrożenia zabezpieczeń została przedstawiona na rysunku 5.



Rysunek 5. Koncepcja wdrożenia WAF jako zintegrowanego elementu infrastruktury sieciowej

Utrzymanie dostępności usług i ochrona przed atakami D/DoS

Utrzymanie dostępności aplikacji świadczących usługi w sieci Internet wymaga zaprojektowania i wdrożenia odpowiednich środków ochrony, m.in.:

- redundancji urządzeń i łączy (HA) oraz lokalizacji,
- akceleracji serwerów (load balancing, SSL offload, caching, itp.),
- nadmiarowej wydajności urządzeń, serwerów i łączy,
- zarządzania zmianami oprogramowania urządzeń, serwerów i aplikacji,
- ochrony przed atakami D/DoS skierowanymi na aplikacje Web.

W przypadku aplikacji z interfejsem Web szczególnie istotnym zagadnieniem projektowym jest ochrona przed atakami D/DoS. Ataki D/DoS skierowane na konkretną aplikację Web wykonywane z wykorzystaniem odpowiednio dobranych zapytań HTTP (GET, POST) potrafią spowodować jej niedostępność za pomocą niewielkiej liczby zapytań, spreparowanych tak, aby maksymalnie angażować CPU i pamięć serwera. Konwencjonalne sieciowe zabezpieczenia firewall i IPS nie mają możliwości zablokowania takich ataków DoS, ponieważ liczba otwieranych sesji nie jest duża i są to legalne zapytania do aplikacji.

Identyfikacja aplikacyjnych ataków D/DoS w systemach WAF jest możliwa poprzez analizę czasu odpowiedzi serwera (opóźnienia) oraz częstości napływających zapytań do serwera. Istotne zwiększenie opóźnienia w odpowiedziach serwera lub częstości zapytań napływających do serwera może wskazywać na atak D/DoS. Analiza ruchu dokonywana jest w odniesieniu do danych historycznych. Ataki D/DoS wykonywane są za pomocą specjalizowanych narzędzi. Nie odbywa się to z wykorzystaniem przeglądarek Web. Skuteczną metodą ochrony jest identyfikacja czy zapytania do aplikacji wysyłane są przez przeglądarkę, czy inne narzędzia. Zabezpieczenia WAF posiadają zaimplementowane metody rozpoznawania czy zapytania wysyłane są przez przeglądarki Web (m.in. dodają kod JavaScript do odpowiedzi serwera i weryfikują kolejne zapytania) i umożliwiają filtrowanie ruchu generowanego przez inne narzędzia.

Konkluzja

Systemy e-commerce, e-banking i większość aplikacji w Centrach Danych zbudowane są z użyciem technologii Web dla zapewnienia ich większej dostępności. Systemy te zwykle świadczą swoje usługi w sposób otwarty poprzez sieć Internet. Przez to narażone są na liczne niebezpieczeństwa, jak włamania i ataki D/DoS. Należące do zaufanych firm, źle zabezpieczone aplikacje Web są powszechnie wykorzystane przez przestępców do dys-

trybucji złośliwego oprogramowania (tzw. drive-by download). Ochrona aplikacji Web jest zagadnieniem złożonym projektowo i technologicznie. Wymaga zastosowania środków bezpieczeństwa adekwatnych do rzeczywistych zagrożeń oraz regularnego audytowania portalu wraz z rozwojem oprogramowania.

W celu odpowiedniego planowania i utrzymania bezpieczeństwa aplikacji Web firmy mogą skorzystać z dostępnych w tym obszarze standardów (m.in.: OWASP ASVS, PCI DSS). Należy przy tym pamiętać, że testy bezpieczeństwa nie dają stuprocentowej pewności wykrycia wszystkich podatności aplikacji Web. Nie zawsze też możliwe jest usunięcie wszystkich wykrytych podatności, w szczególności gdy zostały zidentyfikowane przed samym oddaniem aplikacji do produkcyjnego użycia. Dlatego też oprócz testów bezpieczeństwa wykonanych przez kompetentnego audytora aplikacja Web powinna zostać wyposażona w dedykowane zabezpieczenia WAF (ew. inne specjalizowane zabezpieczenia do ochrony aplikacji Web i baz danych), utrudniające wykorzystanie możliwych podatności.

