Marcin Piosek

Burp Suite Community Edition – wprowadzenie do obsługi proxy HTTP

WSTĘP

Przeprowadźmy wspólnie test bezpieczeństwa aplikacji WWW. Spróbujmy znaleźć podatności, które wykorzystamy do tego, by zmusić aplikację do wykonania operacji, jakie nie zostały przewidziane przez jej autora. Ustalmy również, czy możemy sięgnąć po dane, które nie powinny być dla nas dostępne. Plan brzmi świetnie! Pytanie – jak go zrealizować?

Aplikacje internetowe działają, wykorzystując protokół HTTP[1]. Przeglądarka WWW wysyła do serwera zapytanie, a ten odsyła w odpowiedzi dane, które zawierają np. kod HTML[2], JavaScript[3] oraz CSS[4]. Przeglądarka po odebraniu tych informacji odpowiednio je przetwarza, przy czym otrzymujemy wynik w postaci strony internetowej lub innego zasobu będącego przedmiotem żądania.

Jeżeli chcemy przeprowadzić analizę bezpieczeństwa aplikacji internetowej, musimy znaleźć sposób na przeanalizowanie komunikacji pomiędzy przeglądarką WWW a serwerem. Rozwiązanie jest zaskakująco proste – musimy użyć proxy HTTP! W naszym przypadku będzie nim Burp Suite Community Edition[5], którego w dalszej części tego rozdziału będziemy nazywać Burpem.

WYZNACZAMY CEL

Zapoznanie się z informacjami zamieszczonymi w tym materiale powinno pozwolić osobie, która nie miała wcześniej styczności z analizą bezpieczeństwa aplikacji WWW, na zdobycie podstawowej wiedzy z zakresu działania protokołu HTTP, aplikacji WWW oraz proxy HTTP Burp. Przekazana tu wiedza powinna wystarczyć do tego, by móc samodzielnie przeanalizować sposób działania dowolnej aplikacji. Aby osiągnąć ten cel, omówione zostaną podstawy konfiguracji środowiska audytorskiego związane z ustawieniami przeglądarki WWW oraz podstawy obsługi proxy z wykorzystaniem Burpa*.

CZYM JEST BURP SUITE?

Burp to jedno z dostępnych na rynku narzędzi typu proxy HTTP. Pozwala na przechwytywanie zapytań generowanych przez przeglądarki WWW oraz inne opro-

* Omówienie obsługi proxy nie obejmie opcji zaawansowanych oraz dostępnych w pełnej, płatnej wersji tego narzędzia.

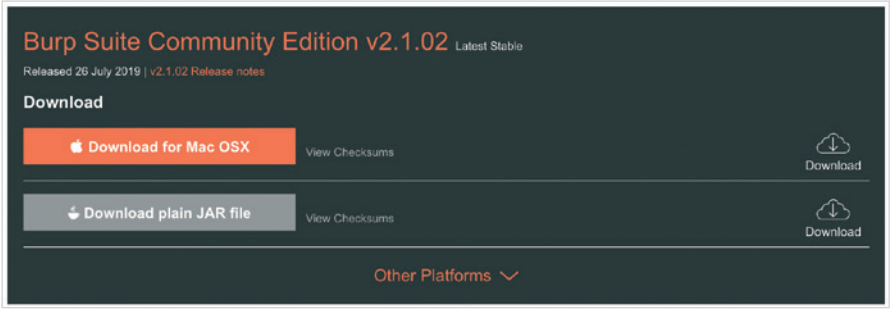
gramowanie, wykorzystujące protokół HTTP. Przechwycenie zapytań za pomocą proxy jest jednoznaczne z możliwością ich dowolnej modyfikacji.

Alternatywy i dlaczego Burp?

Dostępnych jest kilka innych rozwiązań, które dostarczają podobny zestaw funkcji jak Burp. Bezpośrednimi konkurentami są przede wszystkim: Fiddler[6] oraz OWASP ZAP[7]. Za pomocą każdego z nich będziemy w stanie wykonać większość podstawowych zadań, jakie trzeba przeprowadzić w trakcie audytu bezpieczeństwa aplikacji WWW. Wystawienie jednoznacznej oceny i podjęcie decyzji o tym, które z tych narzędzi jest lepsze, wymagałoby rozbudowanej analizy porównawczej. Bazując na moim doświadczeniu, Drogi Czytelniku, mogę powiedzieć, że Burp Suite jest prostym w obsłudze rozwiązaniem nieprzytłaczającym użytkownika mnogością zakładek i pozycji w menu, a z drugiej strony dostarczającym dokładnie tego, co niezbędne do pracy audytora czy pentestera.

Pobranie i uruchomienie Burp Suite Community Edition

Darmowa, również do użytku komercyjnego, wersja Burp Suite dostępna jest na stronie PortSwigger[8] (rysunek 1). Do wyboru mamy pobranie pliku `JAR` lub instalatora na wybraną platformę. Na potrzeby artykułu wystarczy, że skorzystamy z pierwszej opcji, czyli pobierzemy `JAR`.



Rysunek 1. Widok strony z linkami do pobrania instalatora Burp Suite

Po zapisaniu archiwum na dysku warto zweryfikować, czy suma kontrolna pliku `JAR` zgadza się z informacją zamieszczoną na stronie *portswigger.net*.

Listing 1. Obliczenie sumy kontrolnej pobranego pliku na systemach Linux/macOS

```
$ shasum -a256 burpsuite_community_v2.1.02.jar
e9ac253770fe716abee8cd1985494d065e2efd00df0b433187afc1bec508a432  burpsuite_
community_v2.1.02.jar
```

Zanim uruchomimy Burpa, musimy się upewnić, że na naszej stacji roboczej zainstalowane jest oprogramowanie Java Runtime Environment. W tym celu możemy w konsoli uruchomić polecenie `java` z parametrem `-version`.

Listing 2. Wynik weryfikacji instalacji środowiska Java

```
$ java -version
java version "1.8.0_131"
Java(TM) SE Runtime Environment (build 1.8.0_131-b11)
Java HotSpot(TM) 64-Bit Server VM (build 25.131-b11, mixed mode)
```

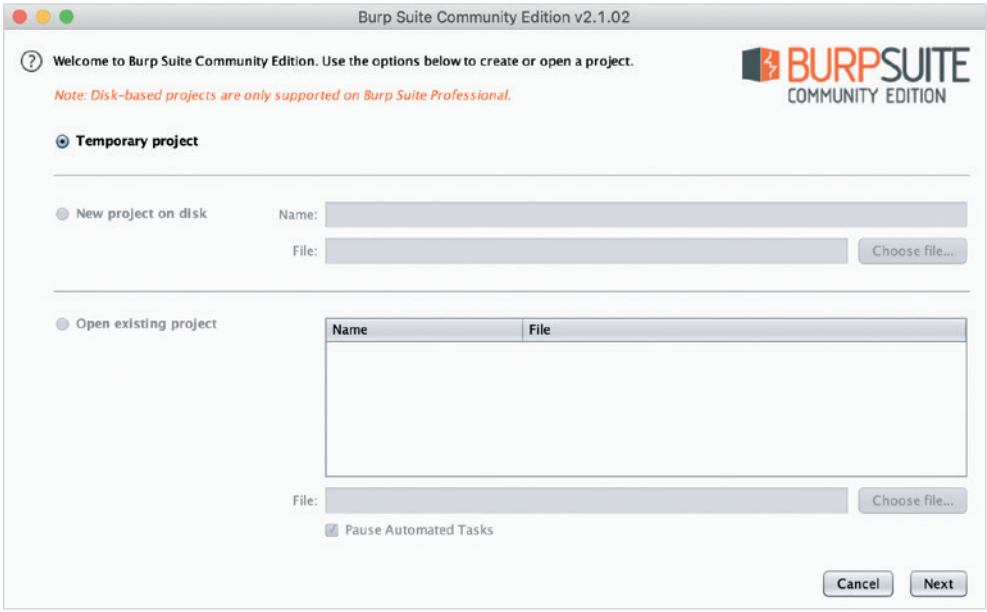
Jeżeli wynik wykonania tego polecenia na naszej stacji nie przypomina tego z listingu 2, wtedy powinniśmy pobrać[9] i zainstalować środowisko uruchomieniowe Java.

Przed uruchomieniem pliku warto jeszcze dla pewności zweryfikować jego bezpieczeństwo, np. z wykorzystaniem serwisu VirusTotal[10].

Jeżeli jesteśmy pewni zawartości pobranego archiwum, możemy uruchomić Burpa za pomocą polecenia:

```
$ java -jar burpsuite_community_v2.1.02.jar
```

Po wydaniu tego polecenia powinien ukazać się ekran powitalny, a chwilę później okienko konfiguracji ustawień projektu (rysunek 2). Ponieważ korzystamy z wersji darmowej, jedyną opcją, jaką możemy wybrać na tym etapie, jest przycisk NEXT.

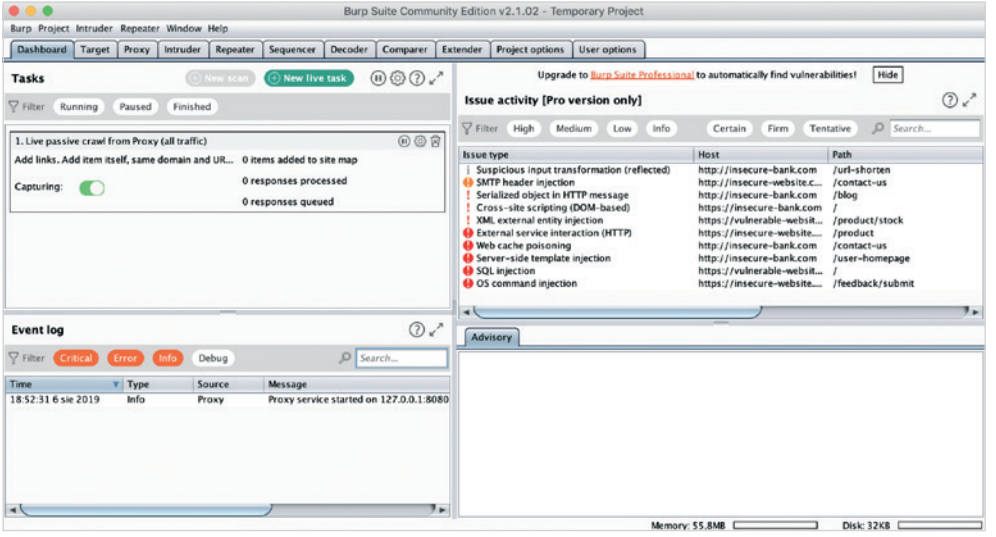


Rysunek 2. Ekran startowy Burp Suite

Kolejne okno udostępnia funkcje, które można wykorzystać do załadowania wcześniej zapisanej konfiguracji tego narzędzia. Jeżeli jednak uruchamiamy Burpa pierwszy raz, powinniśmy zauważyć, że wszystkie listy są puste (rysunek 3).

Jak widać w kolejnym kroku, nie pozostało nam nic innego, jak tylko kliknąć przycisk START BURP. Po chwili wyświetli się okno podobne do tego z rysunku 4. Tak prezentuje się uruchomiony Burp.

Rysunek 3. Widok formularza konfiguracji ustawień projektu w Burp Suite



Rysunek 4. Uruchomiony Burp

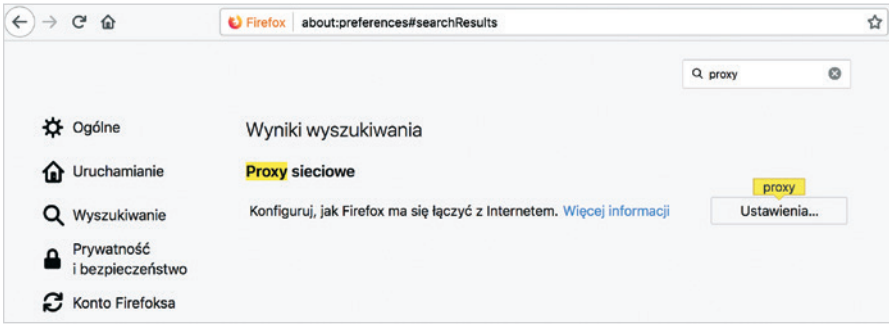
Burp właściwie zaraz po uruchomieniu jest skonfigurowany i gotowy do pracy. Zanim przystąpimy do testów, musimy jeszcze przygotować naszą przeglądarkę, tak by przekazywała zapytania do proxy.

KONFIGURACJA ŚRODOWISKA PRACY

Przeglądarki WWW takie jak Firefox, Chrome czy Internet Explorer nie są domyślnie skonfigurowane do pracy z narzędziami typu Burp. W tym materiale po-

każemy, jak skonfigurować przeglądarkę Firefox, by zapytania w niej generowane były przesyłane do Burpa, a dopiero później „w świat".

W tym celu należy przejść do menu przeglądarki, a następnie wybrać PREFERENCJE / OPCJE. Można też wpisać w pasek przeglądarki adres *about:preferences*.



Rysunek 5. Widok okna konfiguracji przeglądarki

Widok okna ustawień przeglądarki zmienia się w czasie, dlatego najprościej będzie skorzystać z wyszukiwarki, która umieszczona jest w prawym górnym rogu okna (rysunek 5). Po wpisaniu tam słowa „proxy" aplikacja zawęzi dostępne ustawienia tylko do tych, które związane są z wybraną frazą.

W kolejnym kroku powinniśmy wybrać przycisk USTAWIENIA, po czym przeglądarka wyświetli nowe okno (rysunek 6).

Rysunek 6. Konfiguracja ustawień proxy w przeglądarce Mozilla Firefox

Domyślnie zaznaczona będzie opcja BEZ SERWERA PROXY. Aby zapytania wysyłane przez przeglądarkę trafiały do proxy, musimy wybrać opcję RĘCZNA KONFIGURACJA SERWERÓW PROXY, a w polu obok wprowadzić adres 127.0.0.1 oraz port 8080. Taka

konfiguracja oznacza, że po wybraniu przycisku OK, który zapisze zmiany, przeglądarka będzie przekazywała cały ruch HTTP/HTTPS pod adres 127.0.0.1 na port 8080.

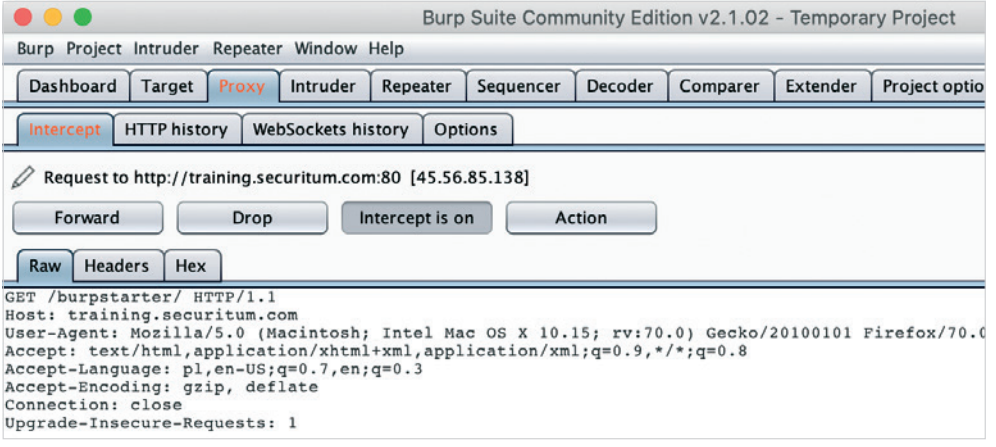
Zanim sprawdzimy, czy wprowadzone przez nas zmiany dały zamierzony skutek, możemy zadać sobie pytanie, dlaczego wpisaliśmy w ustawienia przeglądarki akurat takie, a nie inne wartości. Aby to wyjaśnić, wystarczy, że przejdziemy do Burpa, a następnie do zakładki PROXY i dalej podzakładki OPTIONS. W sekcji PROXY LISTENERS znajdziemy listę, na której wyszczególnione będzie, na jakim interfejsie oraz porcie nasłuchuje nasze proxy. Domyślnie będzie to właśnie interfejs localhosta, stąd adres 127.0.0.1 oraz port 8080.

PRZYSTĘPUJEMY DO PRACY

Na tym etapie powinniśmy mieć uruchomione proxy Burp oraz przeglądarkę WWW skonfigurowaną tak, by przekazywała ruch przez proxy. Przyszedł więc czas, by przechwycić pierwsze zapytanie HTTP. Aby to zrobić, wystarczy, że w przeglądarce WWW przejdziemy pod adres `http://training.securitum.com/burpstarter/`[11]. Dostępna jest tam prosta aplikacja WWW, którą wykorzystamy do poznania możliwości Burpa.

> Początkowo będziemy pracować na wersji aplikacji dostępnej przez nieszyfrowany kanał komunikacji HTTP. Warto zatem zwrócić uwagę, czy w pasku adresu na pewno pojawia się http zamiast https. Informację o tym, jak poradzić sobie z aplikacjami wykorzystującymi HTTPS, zamieszczono w akapicie „Połączenie nie jest bezpieczne – HTTPS".

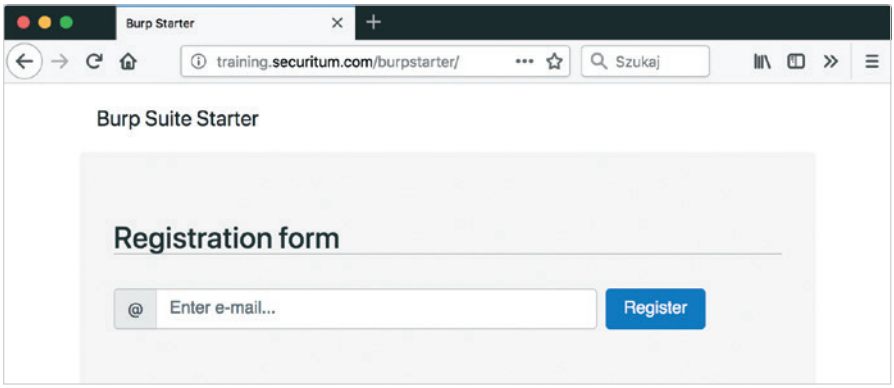
Po przejściu pod podany adres powinniśmy zauważyć, że przeglądarka próbuje połączyć się z aplikacją, ale nie może tego zrobić. Musimy teraz przejść do naszego proxy HTTP. W zakładce INTERCEPT zauważymy, że Burp przechwycił wygenerowane przez przeglądarkę zapytanie HTTP (rysunek 7).



Rysunek 7. Zapytanie HTTP przechwycone przez Burp Suite

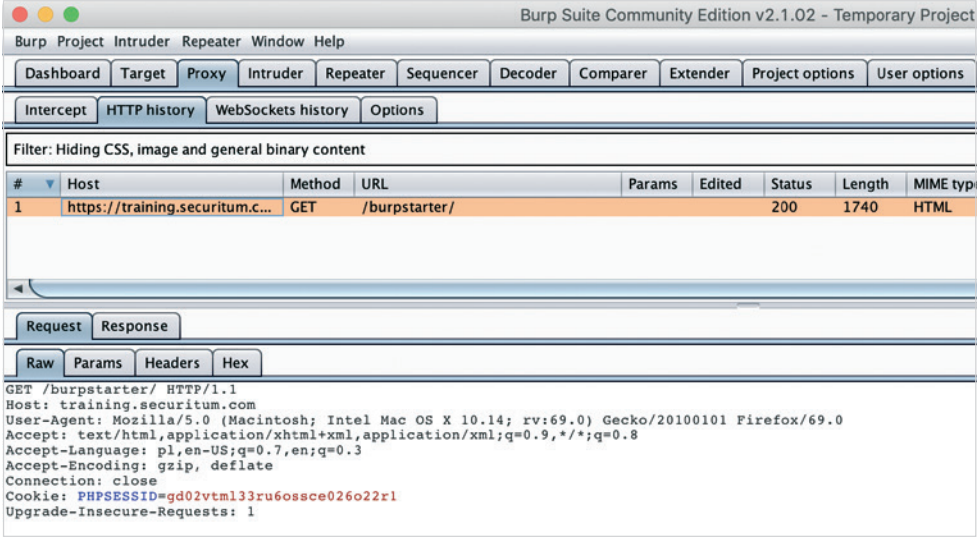
Możemy w tym widoku zauważyć kilka rzeczy. Najważniejszą z nich jest treść zapytania HTTP. Mowa tu o takich danych, jak nazwa metody HTTP `GET` oraz nagłówki HTTP jak `Host`, `User-Agent` czy `Accept`.

Już na tym etapie możemy wprowadzić do zapytania dowolne zmiany, np. zmodyfikować zasób, do którego wykonywane jest zapytanie, lub zmienić wartość czy wręcz usunąć wybrany nagłówek. Wybierzmy na razie opcję FORWARD. Spowoduje to, że proxy prześle zapytanie dalej do serwera, a my po chwili będziemy mogli w przeglądarce zobaczyć wczytaną stronę (rysunek 8).



Rysunek 8. Testowa aplikacja przetworzona przez przeglądarkę

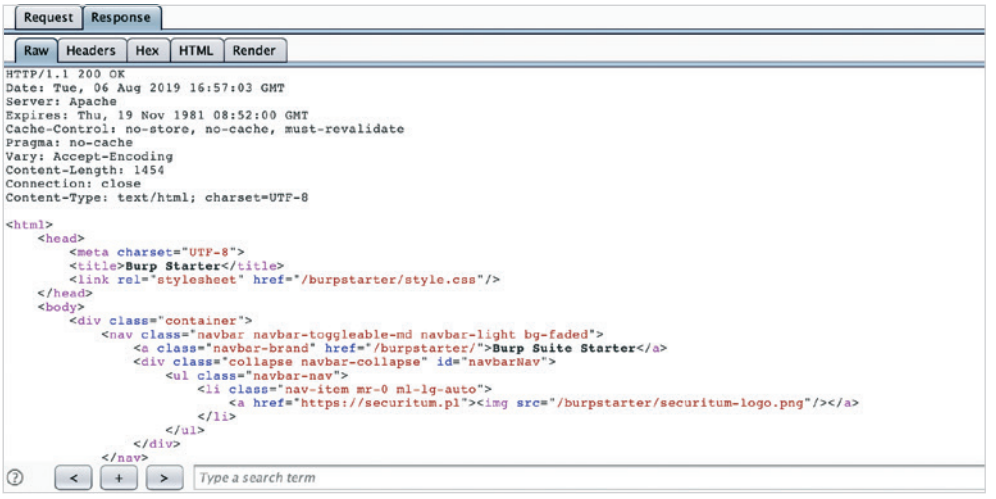
Przechodząc do zakładki HTTP HISTORY, możemy zobaczyć zapis i historię wcześniej wykonywanych zapytań (rysunek 9).



Rysunek 9. Zakładka HTTP HISTORY z listą wszystkich przechwyconych zapytań

Okno historii zapytań składa się z dwóch części. Pierwszą z nich jest lista, na której znajdują się wszystkie przechwycone do tej pory zapytania. Po wybraniu dowolnej pozycji w dolnej części ekranu pojawi się treść wybranego zapytania. Burp prezentuje zarówno zapytanie, w zakładce REQUEST, jak i odpowiedź serwera – w zakładce RESPONSE (rysunek 10).

Oprócz zapytań, które sami wygenerowaliśmy, na liście z zakładki HTTP HISTORY możemy znaleźć inne pozycje. Najczęściej ich źródłem będzie sama przeglądarka, która okresowo sprawdza, czy dostępne są dla niej nowe aktualizacje. Możliwe również, że mamy po prostu otwartą więcej niż jedną kartę w przeglądarce, a w nich załadowane inne aplikacje WWW.



Rysunek 10. Treść odpowiedzi serwera prezentowana przez Burp Suite

Zarówno zapytania, jak i odpowiedzi mogą być przez proxy prezentowane w różnych postaciach. Po przejściu do zakładki REQUEST zobaczymy takie podzakładki, jak:

- RAW – widok, w którym zapytanie prezentowane jest w formie nieprzetworzonej,
- PARAMS – narzędzie zwróci tam jedynie listę parametrów, do których zalicza się parametry przekazane w adresie URL (np. `/?id=1`), parametry przekazane w ciele zapytań typu `POST`, jak również wartości ciasteczek HTTP,
- HEADERS – czasem interesujące nas informacje znajdują się w nagłówkach HTTP, ta zakładka prezentuje jedynie nagłówki, które pojawiły się w zapytaniu HTTP,
- HEX – w przypadku, gdy przesyłamy dane binarne, pomocna może okazać się możliwość prześledzenia zapytania w formie heksadecymalnej; taką opcję znajdziemy w zakładce HEX.

Zakładka RESPONSE również zawiera takie podzakładki, jak RAW, HEADERS czy HEX; spełniają one identyczną funkcję jak te z zakładki REQUEST. Oprócz nich pojawią się tam jednak takie pozycje, jak:

- HTML – widok, w którym Burp zamieści jedynie ciało odpowiedzi HTTP, co w większości przypadków sprowadza się do prezentacji kodu HTML, jaki

zwrócił serwer. Burp dodatkowo formatuje kod, poprawiając m.in. wcięcia, przez co zwiększa się jego czytelność,

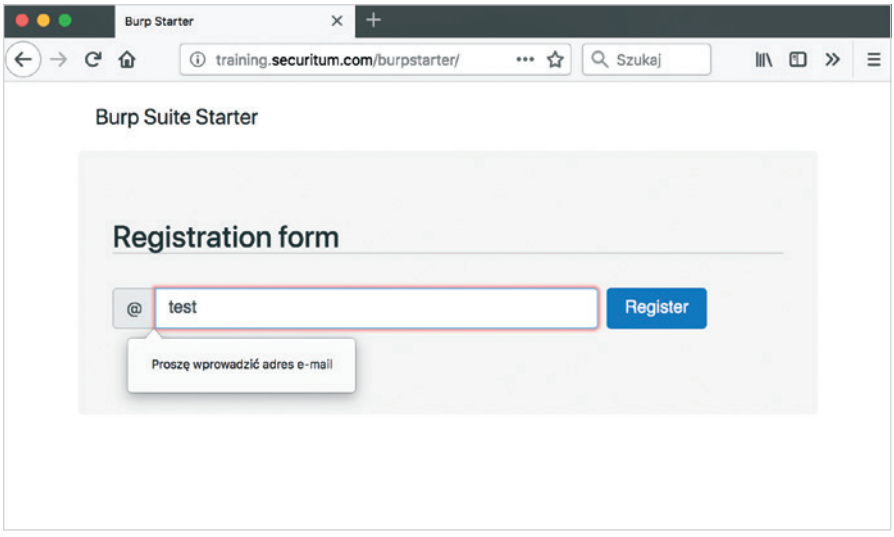
- RENDER – po przejściu do tej zakładki Burp wyzwoli akcję renderowania kodu HTML zwróconego w odpowiedzi HTTP. Można za jej pomocą sprawdzić, jak będzie wyglądać w przeglądarce przetworzony kod HTML. Burp od wersji 2 używa do renderowania strony silnika z przeglądarki Chromium.

MODYFIKOWANIE ZAPYTAŃ HTTP

Jesteśmy już po etapie konfiguracji środowiska pracy, jak i po przechwyceniu pierwszego zapytania. Przyszedł czas, by zmodyfikować wybrane zapytanie HTTP. Powstaje jednak pytanie: do czego może nam to właściwie być potrzebne?

Testy bezpieczeństwa opierają się na założeniach zbliżonych do tych, jakie stosowane są przy „zwykłych" testach aplikacji. Tester bardzo często sprawdza zachowanie aplikacji, np. gdy w polu liczbowym pojawi się litera lub inny ciąg znaków. Kolejnym przykładem podejścia testerów jest podanie na wejściu aplikacji pliku PDF, gdy standardowo powinien pojawić się tam plik CSV. Inaczej mówiąc, weryfikujemy zachowanie aplikacji w przypadku, gdy wymusimy na niej działanie nieprzewidziane przez autora.

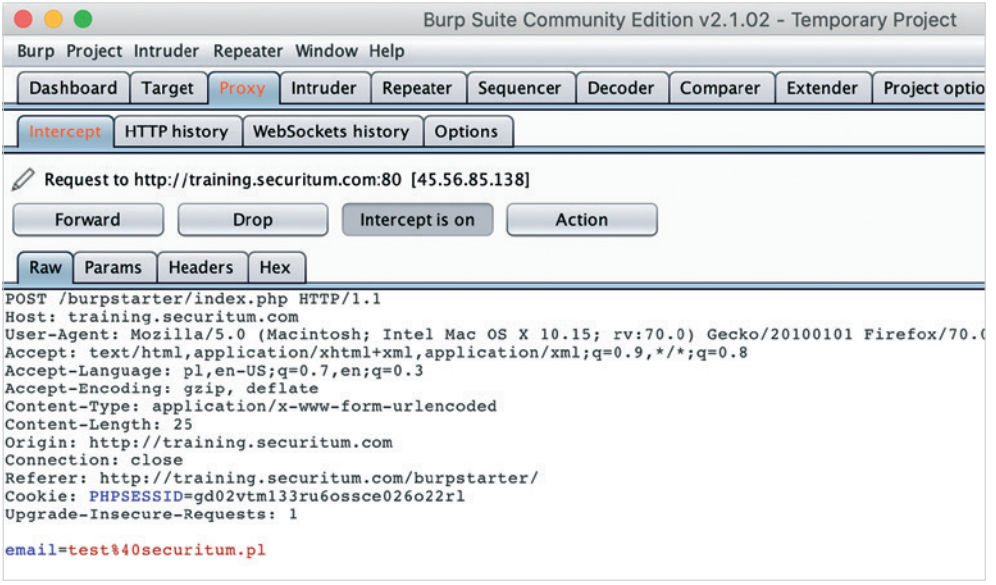
Nasza testowa aplikacja posiada prosty formularz, który zawiera w sobie jedno pole pozwalające na wpisanie adresu e-mail. Jeżeli wprowadzimy tam dowolne dane niezgodne z formatem adresu e-mail, a następnie klikniemy na przycisk REGISTER, przeglądarka nie wygeneruje żadnego zapytania HTTP, za to wyświetli komunikat błędu jak na rysunku 11.



Rysunek 11. Zachowanie testowej aplikacji po wprowadzeniu niepoprawnych danych

Wygląda więc na to, że aplikacja została zabezpieczona przed wprowadzaniem danych w nieodpowiednim formacie. Jako testerzy bezpieczeństwa musimy spróbować przekazać do serwera coś, co adresem e-mail nie jest, i zweryfikować, jak część serwerowa aplikacji zachowa się w takiej sytuacji.

Aby osiągnąć zamierzony cel, na początku wprowadźmy w polu formularza dowolny adres, który będzie poprawny pod względem formatu. Użyję adresu *test@securitum.pl*. Gdy wprowadzimy adres i wybierzemy opcję REGISTER, w proxy Burp powinniśmy zauważyć przechwycone kolejne zapytanie HTTP (rysunek 12).



Rysunek 12. Przechwycone zapytanie HTTP z adresem e-mail

Teraz nie pozostaje nam nic innego, jak tylko zmodyfikować jego treść. Pole, w którym znajduje się zawartość zapytania HTTP, zachowuje się jak standardowy edytor tekstowy. Możemy w nim dowolnie modyfikować treść zapytania. Spróbujmy na początek usunąć tekst `test%40securitum.pl`, czyli wprowadzony przez nas adres e-mail, i zastąpić go ciągiem znaków payloadu: `test<script>alert(document.domain);</script>`.

> Twoją uwagę, Drogi Czytelniku, mógł przykuć sposób zapisu adresu e-mail w formie `test%40securitum.pl`, a szczególnie ciąg znaków `%40`. Taki, a nie inny sposób przedstawienia znaku @ wynika z tego, że zgodnie z RFC 3986[12] znak ten należy do grupy zarezerwowanych, a co za tym idzie, po zastosowaniu tzw. kodowania procentowego (ang. *percent-encoding*)[13], przedstawiany jest w formie ciągu znaków rozpoczynającego się od znaku procenta, po którym następuje heksadecymalna reprezentacja danego symbolu. W przypadku @ jest to wartość 40.

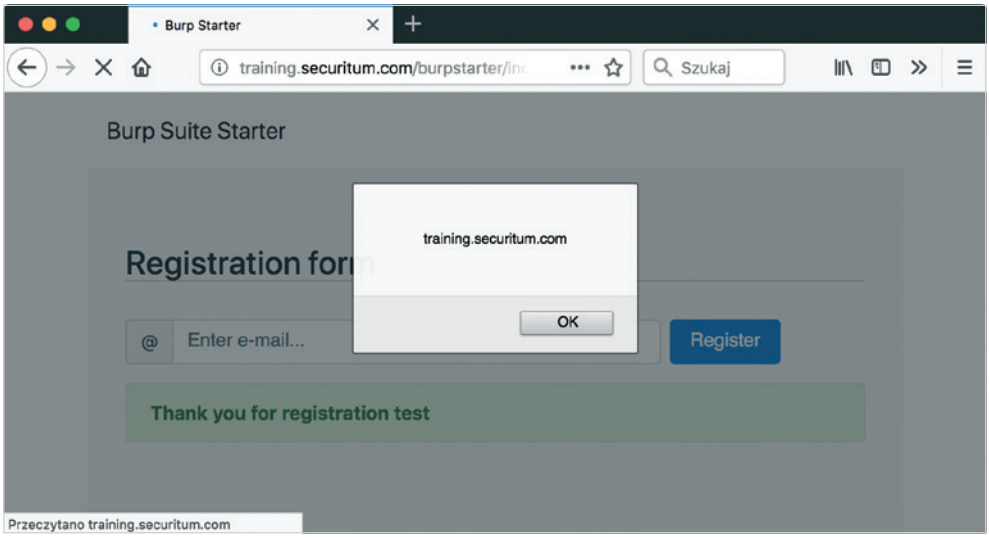
Efekt modyfikacji powinien być podobny do tego z rysunku 13.

```
POST /burpstarter/index.php HTTP/1.1
Host: training.securitum.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.13; rv:62.0) Gecko/20100101
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: pl,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://training.securitum.com/burpstarter/
Content-Type: application/x-www-form-urlencoded
Content-Length: 25
Cookie: PHPSESSID=fturpbtonjqlbq5kc2ejhcfd17
Connection: close
Upgrade-Insecure-Requests: 1

email=test<script>alert(document.domain);</script>
```

Rysunek 13. Zmodyfikowane zapytanie HTTP

Jeżeli wprowadziliśmy zmiany, możemy je zaakceptować przyciskiem FORWARD. Po powrocie do przeglądarki będzie na nas czekać widok podobny do tego z rysunku 14.



Rysunek 14. Efekt przesłania do serwera zmodyfikowanego zapytania HTTP

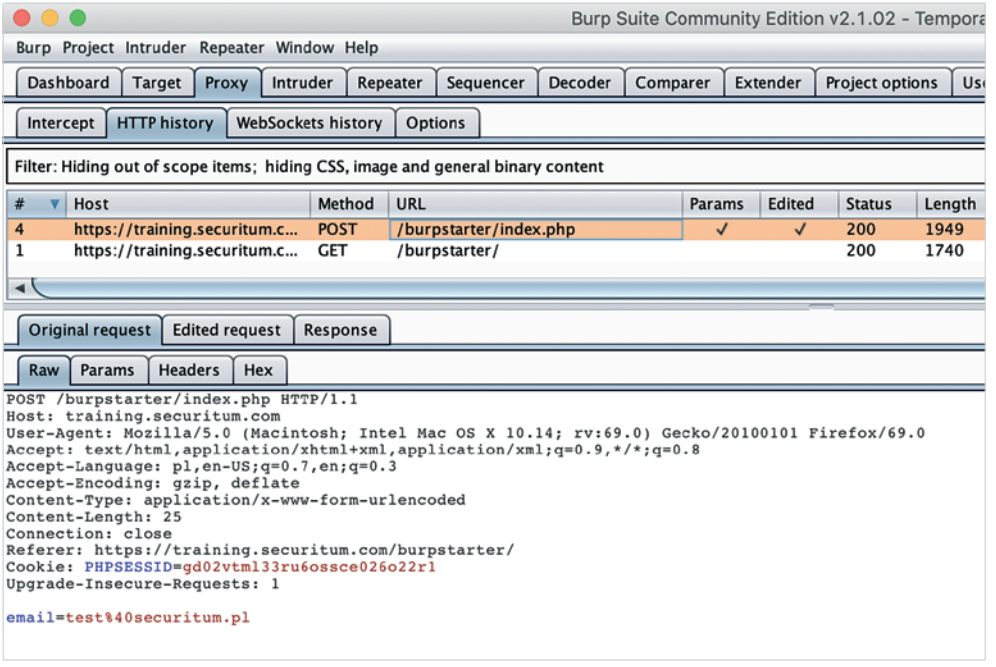
Przyszedł czas, aby przeanalizować, co się właściwie stało. Na pierwszym planie możemy zauważyć nie aplikację, a okienko z komunikatem, w którego treści znajduje się nazwa domeny *training.securitum.com*. Jest to oczywiście związane wprost z ciągiem znaków, jaki wprowadziliśmy w Burpie zamiast adresu e-mail. Okazuje się, że aplikacja waliduje poprawność adresu e-mail jedynie w przeglądarce! Zmodyfikowanie zapytania w proxy i wysłanie go do serwera skutkuje tym, że aplikacja w odpowiedzi zwróci wprowadzony przez nas adres. To z kolei wprost prowadzi do podatności *Cross-Site Scripting*[*], w skrócie XSS. W tym miejscu możemy odnotować pierwszy mały sukces. Właśnie wykryliśmy podatność bezpieczeństwa w aplikacji WWW!

* Więcej o tej podatności w rozdz. *Podatność Cross-Site Scripting (XSS)*.

Przeprowadzony przez nas eksperyment pokazuje, jak ważne jest zrozumienie, że zabezpieczenie, w tym walidacja danych po stronie klienta, może być traktowane jedynie jako dodatek. Właściwy i konieczny do zaimplementowania mechanizm walidacji danych powinien znajdować się w części serwerowej aplikacji.

Możemy jeszcze sprawdzić, jak doszło do tego, że aplikacja wyświetliła alert JavaScript zamiast adresu e-mail. Prześledźmy, jak wyglądało oryginalne zapytanie, następnie jego wersja zmodyfikowana i odpowiedź serwera. W tym celu musimy ponownie przejść do zakładki HTTP HISTORY i odnaleźć na liście zapytań to, które wygenerowaliśmy po kliknięciu w przycisk REGISTER. Jeżeli zapytań będzie więcej, możemy się zasugerować m.in. tym, że w kolumnie METHOD powinno znajdować się słowo POST.

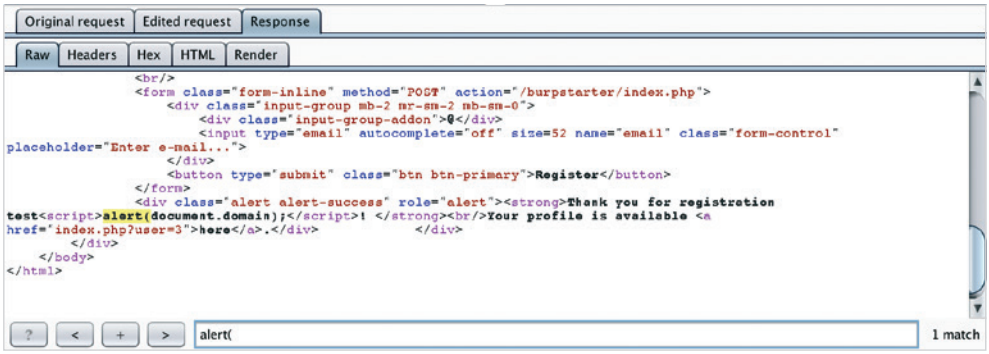
Gdy namierzymy nasze zapytanie, Burp powinien wyświetlić widok podobny do tego z rysunku 15.



Rysunek 15. Zmodyfikowane zapytanie HTTP – widok w zakładce HTTP HISTORY

W drugiej części ekranu możemy zauważyć znaną nam zakładkę RESPONSE, a także dwie inne – ORIGINAL REQUEST oraz EDITED REQUEST. Nazwy są dość jednoznaczne i odpowiadają kolejno: oryginalnej wersji zapytania (z adresem e-mail) oraz wersji zmodyfikowanej, czyli tej, która została przesłana do serwera. Przechodząc do zakładki EDITED REQUEST, możemy przekonać się, jak wyglądało wysłane zapytanie.

Na moment wróćmy jednak do zakładki RESPONSE. W dolnej części ekranu znajdziemy pole, które umożliwia nam wyszukiwanie tekstu w odpowiedzi HTTP. Możemy w ten prosty sposób ustalić, czy wprowadzone przez nas dane na wejściu – np. w parametrze – zostały zwrócone w odpowiedzi HTTP (rysunek 16).

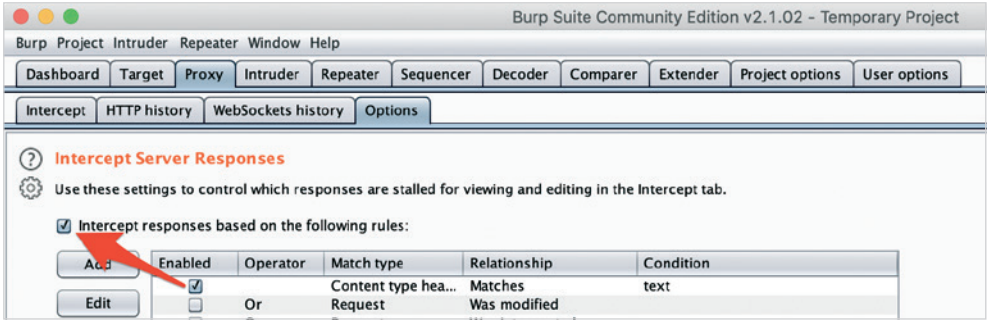


Rysunek 16. Filtrowanie treści odpowiedzi HTTP pod kątem określonej frazy

Jak można zauważyć na rysunku 16, wprowadzone przez nas dane (payload) zostały przez aplikację zwrócone w odpowiedzi HTTP. W efekcie kod JavaScript wprowadzony jako zawartość parametru `email` został przez przeglądarkę potraktowany jako fragment właściwego kodu strony i wykonany.

Przechwytywanie odpowiedzi

Domyślnie Burp przechwytuje jedynie zapytania, które mają zostać przesłane do serwera. Czasem może się jednak zdarzyć, że będzie nam zależało również na modyfikacji odpowiedzi, którą przesyła serwer. Aby móc to zrobić, musimy przejść do zakładki PROXY, następnie OPTIONS i w sekcji INTERCEPT SERVER RESPONSES zaznaczyć opcję INTERCEPT RESPONSES BASED ON THE FOLLOWING RULES (rysunek 17).



Rysunek 17. Konfiguracja Burpa pozwalająca na przechwytywanie odpowiedzi HTTP

Od teraz w zakładce INTERCEPT oprócz zapytań będziemy mogli również modyfikować odpowiedzi zwracane przez serwer.

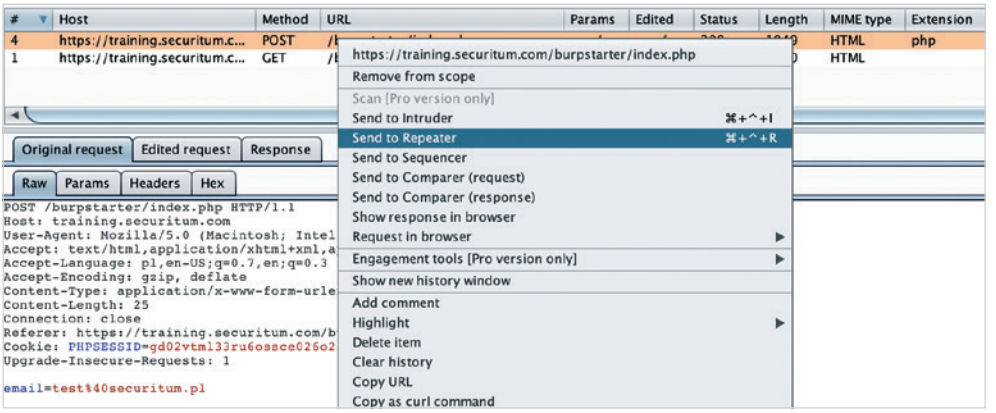
ĆWICZENIE

Przechwyć i zmodyfikuj zapytanie do aplikacji tak, by w komunikacie alertu zamiast nazwy domeny pojawiły się ciasteczka, które przeglądarka zapisała dla domeny *training.securitum.com*.

REPEATER – POWTÓRZ TO JESZCZE RAZ

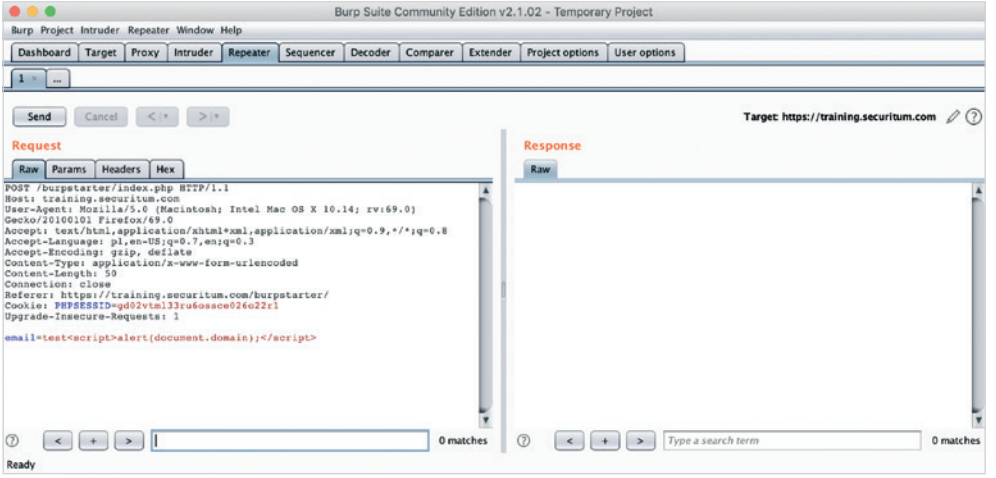
Do tej pory przechwytywaliśmy zapytania po to, by je podejrzeć lub zmodyfikować, a następnie przesłać do serwera. Cała operacja wymagała wyzwolenia zapytania w przeglądarce, a później wykonania wymaganych operacji w proxy. Co jednak, jeżeli chcemy określone zapytanie wysłać do serwera kilkukrotnie, bez konieczności ciągłego wspomagania się przeglądarką? Odpowiedź znajdziemy w zakładce REPEATER.

Przygotujmy środowisko pracy. W tym celu w zakładce HTTP HISTORY kliknijmy prawym przyciskiem myszy na pozycję, która zawiera modyfikowane przez nas zapytanie POST (rysunek 18).



Rysunek 18. Opcja pozwalająca wysłać zapytanie do narzędzia Repeater

Następnie z menu wybierzmy opcję SEND TO REPEATER. Po wykonaniu tej czynności możemy w Burpie przejść do zakładki REPEATER (rysunek 19).

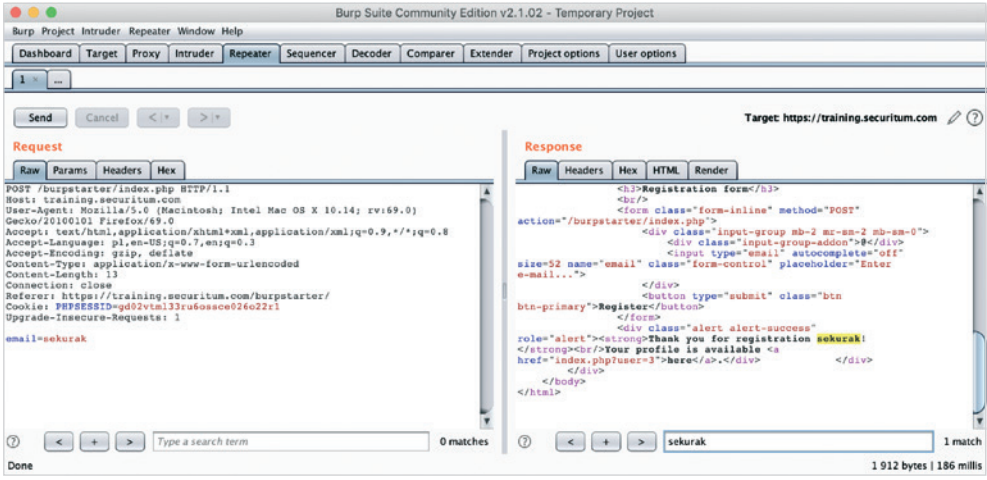


Rysunek 19. Zapytanie HTTP załadowane do narzędzia Repeater

Główne okno tego narzędzia podzielone jest na dwie części. W pierwszej prezentowana jest treść zapytania, w drugiej będziemy mieli dostęp do odpowiedzi, jaką zwraca serwer.

Spróbujmy teraz ponownie zmodyfikować zapytanie, tym razem właśnie w Repeaterze. Edycja treści zapytania odbywa się w taki sam sposób jak poprzednio. Pole tekstowe pozwala nam modyfikować treść zapytania tak jak tekst w dowolnym edytorze. Aby sprawdzić, jak Repeater zachowuje się w praktyce, zmieńmy wartość parametru `email` na dowolny inny tekst. Akcję wysyłania do serwera wyzwalamy przyciskiem SEND.

Po chwili w drugiej części okna powinniśmy zauważyć odpowiedź, jaką wygenerował serwer (rysunek 20). Korzystając z pola wyszukiwania danych, możemy sprawdzić, czy na pewno wprowadzony przez nas ciąg znaków w parametrze `email` pojawił się w odpowiedzi.



Rysunek 20. Odpowiedź HTTP wyświetlona w narzędziu Repeater

Na pierwszy rzut oka Repeater wydaje się dość prostym narzędziem, jednak nie należy go lekceważyć. Praktyka pokazuje, że może to być jedno z narzędzi, z którym będziemy spędzać najwięcej czasu, testując różne payloady i próbując obejść zabezpieczenia aplikacji. Warto poznać inne opcje Repeatera dostępne po kliknięciu prawym przyciskiem myszy na polu tekstowym (rysunek 21). Niektóre z nich poznamy w dalszej części tego rozdziału.

Jak już wspomniałem, Repeater będzie jednym z najczęściej wykorzystywanych narzędzi podczas testów bezpieczeństwa. Po pewnym czasie uciążliwe może stać się ciągłe wybieranie przycisku SEND za pomocą myszy*. Burp pozwala zdefiniować skróty klawiszowe znacznie przyspieszające wysyłanie do serwera kolejnych zapytań.

* Od wersji 2.0 Burpa dla tej akcji można używać domyślnego wbudowanego skrótu: CTRL+spacja.

Rysunek 21. Menu kontekstowe dostępne w narzędziu Repeater

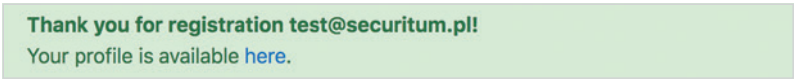
ĆWICZENIE

Przećwicz pracę z narzędziem Repeater. Zweryfikuj, jak zmiany wprowadzane z wykorzystaniem zakładek PARAMS oraz HEADERS wpływają na dane wyświetlane w zakładce RAW. Dodaj do zapytania jeszcze jeden parametr o nazwie `sekurak` i dowolnej wartości, a następnie sprawdź, czy będzie to miało wpływ na działanie aplikacji.

INTRUDER – AUTOMATYZACJA I OSZCZĘDNOŚĆ CZASU

Wyobraźmy sobie, że musimy wielokrotnie wysłać określone zapytanie do serwera. Możemy to robić ręcznie, wspomagając się narzędziem Repeater, ale takie podejście jest możliwe do zaakceptowania tylko w sytuacji, gdy liczba zapytań do wykonania mieści się w granicach kilku lub kilkunastu. Co zrobić, jeżeli są ich tysiące?

Wróćmy jeszcze do naszej testowej aplikacji. Po wpisaniu poprawnego adresu e-mail i wybraniu opcji REGISTER wyświetla ona link do utworzonego profilu użytkownika (rysunek 22).

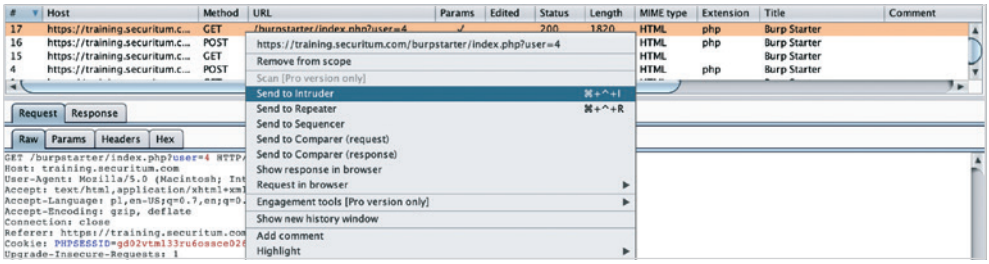


Rysunek 22. Komunikat wyświetlany przez aplikację

Po kliknięciu w link możemy zaobserwować, że zostaliśmy przekierowani pod adres zbliżony do poniższego: *http://training.securitum.com/burpstarter/index.php?user=4*. URL zawiera parametr o nazwie `user`, który definiuje identyfikator zarejestrowanego użytkownika. Gdy patrzymy na takie zachowanie okiem testera bezpieczeństwa, powinna nam się zapalić czerwona lampka sugerująca podatność związaną z możliwością enumeracji danych. Podatność ta polega na przekazywaniu do aplikacji kolejnych identyfikatorów, tak by uzyskać nieautoryzowany dostęp do danych. Zdarza się, że słyszymy o podobnych błędach wykrytych w rzeczywistych systemach. Znane są przypadki, gdy zmiana identyfikatora faktury pozwoliła na dostęp do danych innych podmiotów.

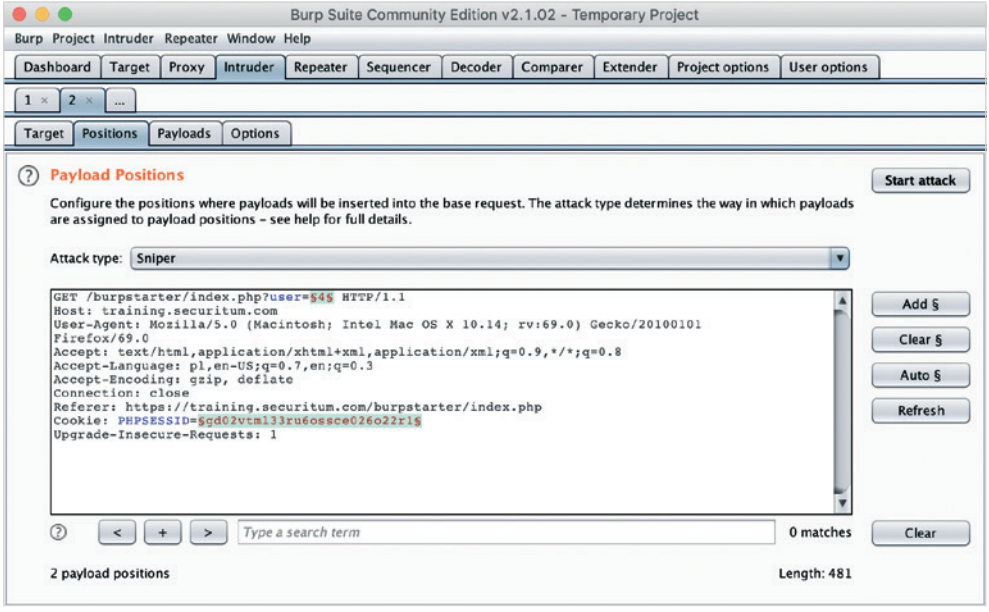
Możemy sprawdzić, czy testowa aplikacja wykorzystywana do nauki obsługi Burpa podatna jest również na enumerację danych. Ustalimy, czy możemy uzyskać dostęp do informacji, które powinny być lepiej chronione. Możemy oczywiście ręcznie modyfikować wartość parametru `user`, wprowadzając tam kolejne liczby całkowite, ale będzie to żmudne zadanie. Opisany przypadek wydaje się idealny dla narzędzia Intruder, które dostępne jest w jednej z zakładek Burpa.

Zanim wykorzystamy Intrudera do enumeracji, musimy go zasilić danymi – w tym przypadku będzie to zapytanie HTTP. Aby to zrobić, przejdźmy ponownie do znanej nam zakładki HTTP HISTORY. Wybieramy interesujące nas zapytanie z listy, następnie z menu kontekstowego – opcję SEND TO INTRUDER (rysunek 23).



Rysunek 23. Opcja wysyłania do narzędzia Intruder

Gdy to zrobimy, możemy przejść do Intrudera. Znajdziemy tam wybrane przez nas zapytanie HTTP (rysunek 24).



Rysunek 24. Zapytanie HTTP prezentowane w narzędziu Intruder

Poznanie możliwości Intrudera w podstawowym zakresie będzie od nas wymagało zapoznania się z opcjami dostępnymi w podzakładkach POSITIONS oraz PAYLOADS.

Zakładka POSITIONS służy do manipulacji zapytaniem, które będziemy wysyłać do serwera. To tutaj określamy, jakie parametry mają być modyfikowane. Aby wskazać, które miejsce zapytania ma być zmieniane, wystarczy zaznaczyć wybrany fragment tak samo jak tekst w dowolnym edytorze tekstu. Następnie wybieramy przycisk ADD, umieszczony z prawej strony edytora. Zanim przejdziemy do zakładki PAYLOADS, przygotujmy zapytanie, aby było gotowe do przeprowadzenia ataku enumeracji użytkowników.

W tym celu wybieramy przycisk CLEAR §. Spowoduje to usunięcie markerów z parametrów, które Burp wykrył automatycznie. Teraz ponownie zaznaczmy wartość parametru `user` – w naszym przypadku wartość 4 – i kliknijmy przycisk ADD §. Efektem tego działania będzie dodanie markera przed i za wspomnianą liczbą. Finalnie powinniśmy osiągnąć efekt jak na rysunku 25.

```
GET /burpstarter/index.php?user=§4§ HTTP/1.1
Host: training.securitum.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.14; rv:69.0) Gecko/20100101
Firefox/69.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: pl,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Connection: close
Referer: https://training.securitum.com/burpstarter/index.php
Cookie: PHPSESSID=gd02vtml33ru6ossce026o22r1
Upgrade-Insecure-Requests: 1
```

Rysunek 25. Poprawna konfiguracja zapytania w narzędziu Intruder

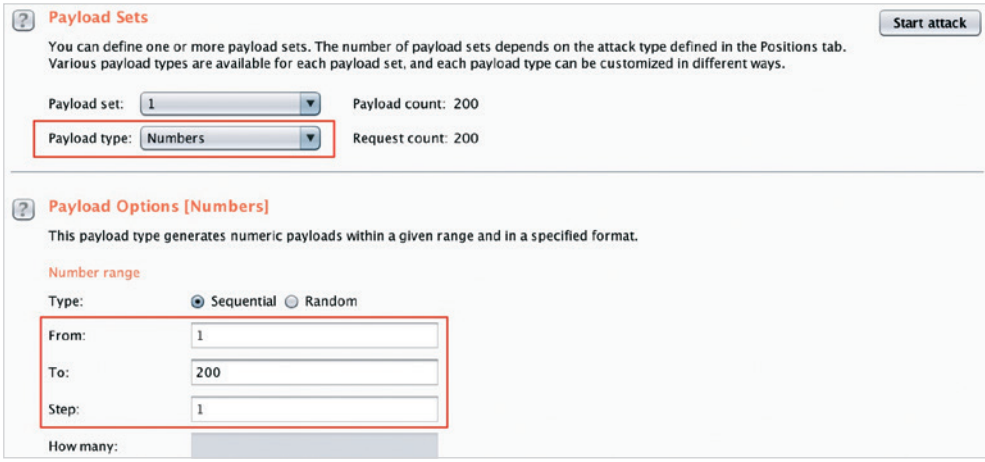
Teraz możemy przejść do zakładki PAYLOADS. Wcześniej zdefiniowaliśmy, które fragmenty zapytania mają być modyfikowane. Pozostało jeszcze wskazanie, co Burp ma wstawić w wybrane przez nas miejsca. Zakładka PAYLOADS umożliwia nam dowolne określenie tego, co chcemy umieścić w zapytaniu. Wśród dostępnych typów payloadów możemy znaleźć m.in. takie pozycje, jak:

- `Simple list` – typ payloadu, w którym wskazujemy po prostu listę ciągów znaków, które kolejno mają być wysłane do aplikacji – dane możemy wprowadzić ręcznie lub wczytać z pliku tekstowego,
- `Numbers` – ten typ będziemy wybierać najczęściej w przypadku, gdy chcemy przeprowadzić enumerację zasobów, odczytywanych przez aplikację po ich identyfikatorach liczbowych – podajemy interesujący nas przedział wartości oraz długość kroku, o jaki ma być zwiększany licznik (ten typ wykorzystamy w naszej testowej aplikacji),
- `Bit flipper` oraz `ECB block shuffler` – typy payloadów, które mogą być nieocenione w przypadku weryfikacji podatności związanych z błędami kryptograficznymi.

Wyżej wymienione typy payloadów to wąska grupa tych, które mogą być wykorzystywane przez nas najczęściej.

Zadaniem, jakie przed nami stoi, jest przeprowadzenie enumeracji użytkowników na podstawie obserwacji tego, że aplikacja w adresie URL przyjmuje jeden parametr, którego wartością są kolejne liczby całkowite. Wygląda więc na to, że najlepszym

wyborem będzie dla nas typ payloadu `Numbers`. Wybierzmy zatem pozycję NUMBERS z listy PAYLOAD TYPE, a następnie zajmijmy się dalszą konfiguracją (rysunek 26).



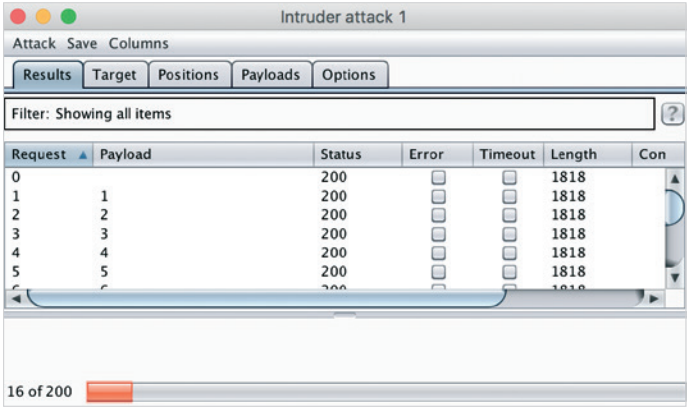
Rysunek 26. Wykorzystywana konfiguracja narzędzia Intruder

Gdy wybierzemy odpowiedni typ payloadu, musimy jeszcze go skonfigurować. W przypadku payloadu `Numbers` powinniśmy wskazać trzy wartości:

- pole FROM – wartość początkowa interesującego nas zakresu,
- pole TO – wartość końcowa,
- pole STEP – krok, o jaki zwiększamy licznik.

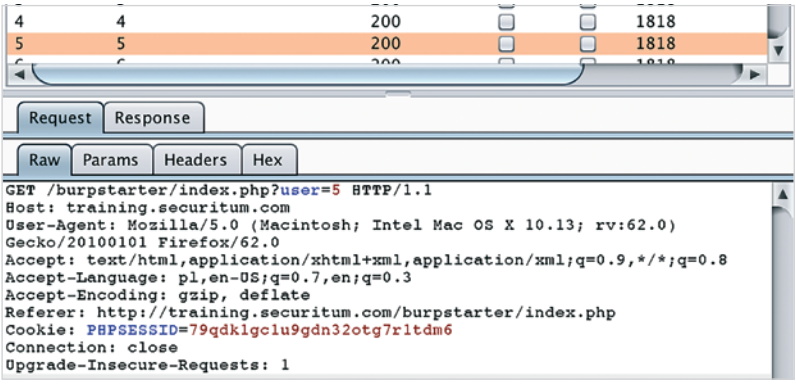
Jeżeli wybraliśmy wszystkie opcje tak jak na rysunku 26, przystępujemy do enumeracji. W celu uruchomienia Intrudera wybieramy przycisk START ATTACK, dostępny w każdej z zakładek w prawym górnym rogu okna.

Po wybraniu tej opcji aplikacja wyświetli komunikat związany z ograniczeniami darmowej wersji Burpa. Po kliknięciu OK pojawi się nowe okno, w którym będziemy mogli śledzić postępy ataku (rysunek 27).



Rysunek 27. Okno śledzenia postępu prac Intrudera

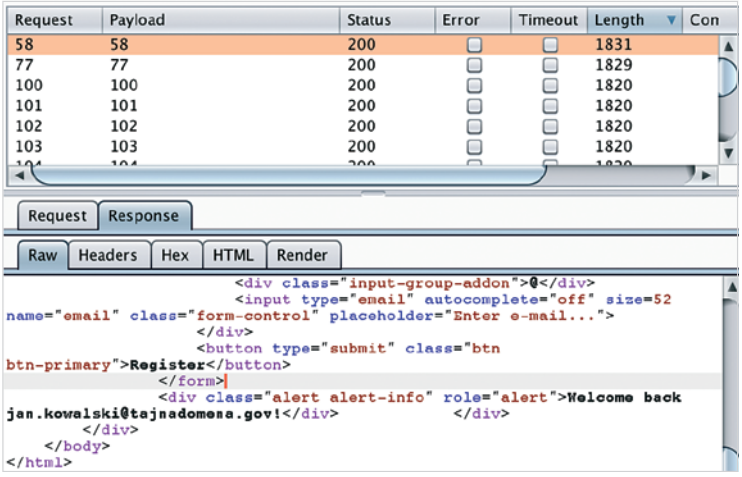
Na dolnej belce nowego okna możemy śledzić postęp wykonania ataku. Główna część okna jest zorganizowana w podobny sposób jak zakładka HTTP HISTORY. Intruder tworzy listę wysłanych zapytań, tak że wybierając dowolną pozycję z listy, możemy sprawdzić, jak zmodyfikowane zostało oryginalne zapytanie i czy w zapytaniu znalazł się payload, który, jak zakładamy, powinien się tam znaleźć. Na przykład: pozycja nr 5 na liście zawiera zapytanie jak to z rysunku 28.



Rysunek 28. Podgląd zapytania zmodyfikowanego przez Intrudera

Widzimy więc, że zgodnie z założeniami Burp wstawia w wysyłane zapytania kolejne liczby całkowite. Poczekajmy chwilę i sprawdźmy, czy przeprowadzając w ten sposób enumerację, uzyskamy dostęp do profili innych użytkowników aplikacji.

Ze względu na ograniczenia darmowej wersji cały proces może zająć nawet kilka minut. Jeżeli wytrwamy do końca, możemy spróbować posortować otrzymane wyniki, np. po długości odpowiedzi, którą reprezentuje kolumna LENGTH. Klikając na nią dwukrotnie, sprawimy, że na samej górze znajdą się te zapytania, dla których serwer zwróci najdłuższą odpowiedź (rysunek 29).



Rysunek 29. Wynik enumeracji danych

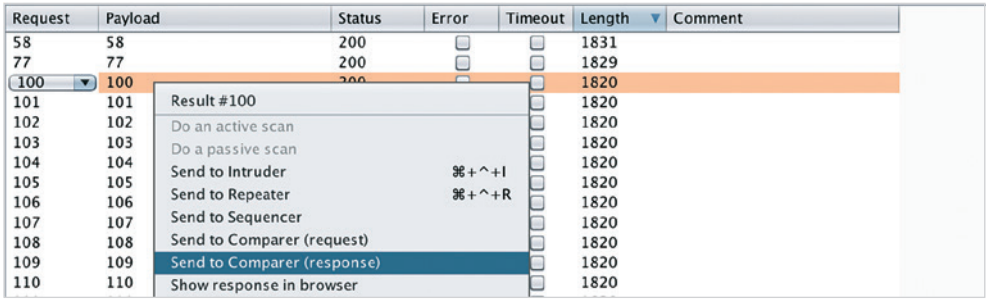
Jeżeli wszystko poszło zgodnie z planem, po posortowaniu pozycji na samej górze listy powinno znaleźć się zapytanie nr 58. Po wybraniu go z listy, a następnie przejściu do zakładki RESPONSE możemy znaleźć pośród kodu HTML adres e-mail innego użytkownika aplikacji. Wygląda na to, że udało nam się przeprowadzić enumerację!

ĆWICZENIE

Wykorzystaj Intrudera jako skaner zasobów (katalogów) na serwerze. Zmodyfikuj zapytanie tak, aby Intruder sprawdzał, czy na serwerze istnieje zasób o określonej nazwie. Listę popularnych nazw katalogów możesz znaleźć na stronie Kali Linux[14].

COMPARER – WSKAŻ RÓŻNICE

Burp wyposażony jest w szeroki zestaw rozmaitych narzędzi. Udostępnia m.in. wygodne narzędzie pozwalające na proste wyszukiwanie różnic pomiędzy wybranymi przez nas zestawami danych. Aby to zrobić, musimy najpierw zasilić narzędzie Comparer, podobnie jak w przypadku Intrudera. Jeżeli nie zamknęliśmy jeszcze okna z wynikiem ataku Intrudera, możemy wykorzystać znajdujące się tam zapytania. Prawym przyciskiem myszy wskażemy dowolne zapytanie. W moim przypadku będzie to pozycja nr 100.

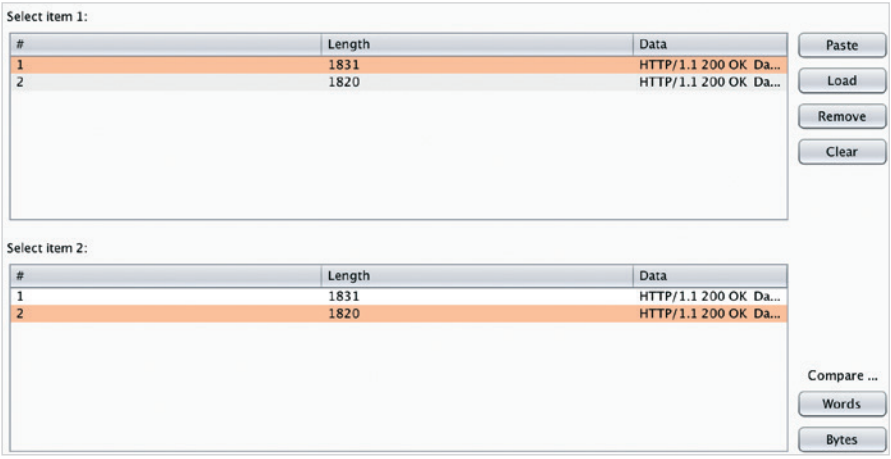


Rysunek 30. Funkcja służąca do przesłania odpowiedzi HTTP do narzędzia Comparer

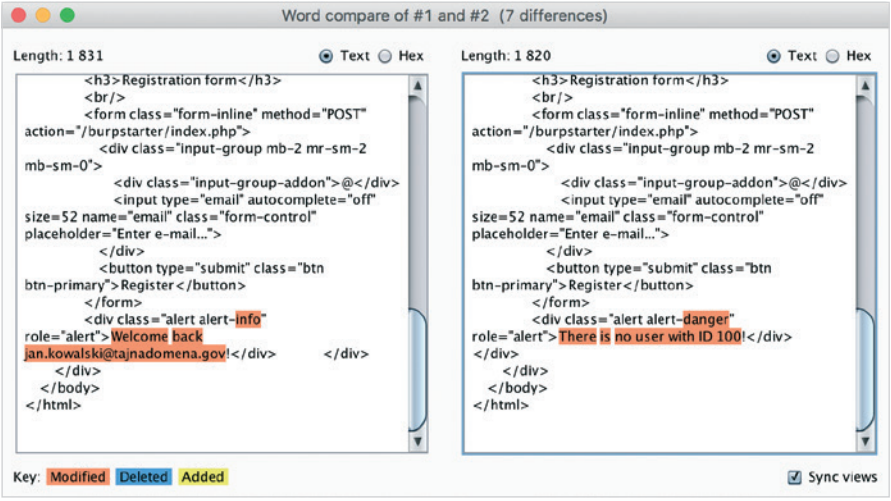
Po rozwinięciu menu kontekstowego musimy wybrać opcję SEND TO COMPARER (RESPONSE) (rysunek 30). Automatycznie zostaniemy przeniesieni do zakładki COMPARER. Widok powinien być podobny do tego z rysunku 31.

Okno Comparera składa się z dwóch list. Na każdej z nich wskazujemy jeden element z pary, która ma być porównana, ponieważ w tym samym czasie możemy zasilić Comparera więcej niż dwoma zapytaniami. Takie rozwiązanie pozwala na wygodne wskazanie, które z nich mają zostać porównane.

Gdy wybraliśmy już dane do porównania, wystarczy, że klikniemy przycisk WORDS. Po chwili Comparer wyświetli nowe okno z wynikami porównania (rysunek 32).



Rysunek 31. Widok okna Comparer po zasileniu narzędzia w dane



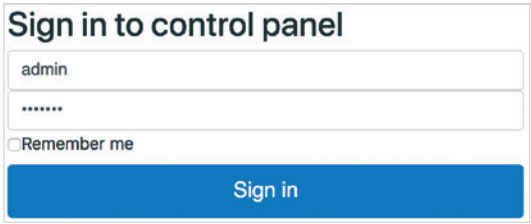
Rysunek 32. Widok porównania odpowiedzi HTTP przez narzędzie Comparer

Zanim przejdziemy do analizy wyników, warto jeszcze zaznaczyć opcję SYNC VIEWS. Dzięki temu, gdy przesuwamy suwak w jednym z pól tekstowych, drugie będzie podążało za pozycją pierwszego. Jak widać na rysunku 32, Comparer porównał dwie odpowiedzi HTTP i w przejrzysty sposób zaprezentował różnice pomiędzy nimi.

DECODER – RADZIMY SOBIE Z „DZIWNYM" CIĄGIEM ZNAKÓW

Do poznawania kolejnych funkcji Burp Suite wykorzystamy inną aplikację, która dostępna jest pod adresem *http://training.securitum.com/burpstarter/admin_panel/*. Zarówno adres URL, jak i to, co wyświetla aplikacja, sugeruje, że jest to formularz logowania do panelu administracyjnego. Spróbujmy teraz zweryfikować, czy jesteśmy w stanie znaleźć w nim podatności bezpieczeństwa.

W formularz logowania wprowadźmy dowolne dane. Na potrzeby przykładu użyję loginu `admin` oraz hasła `sekurak` (rysunek 33).



Rysunek 33. Formularz logowania do panelu administracyjnego uzupełniony o testowe dane

Następnie wybierzmy przycisk SIGN IN. Aplikacja po chwili zwróci komunikat z informacją o niepoprawnych poświadczeniach. Sprawdźmy, jak wyglądało zapytanie, które zostało wyzwolone poprzez wybranie przycisku SIGN IN. Możemy to zrobić, przechodząc do zakładki HTTP HISTORY. Na liście zapytań powinniśmy znaleźć jedno, podobne do tego z listingu 3.

Listing 3. Zapytanie wysłane po uzupełnieniu formularza logowania do panelu administracyjnego

```
POST /burpstarter/admin_panel/ HTTP/1.1
Host: training.securitum.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.13; rv:62.0) ⤶
Gecko/20100101 Firefox/62.0
Accept: */*
Accept-Language: pl,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://training.securitum.com/burpstarter/admin_panel
content-type: application/json
origin: http://training.securitum.com
Content-Length: 40
Cookie: PHPSESSID=891791
Connection: close

{"login":"admin","passw":"c2VrdXJhaw=="}
```

Naszą uwagę powinno przykuć ciało zapytania, a konkretnie – dane w formacie JSON oraz wartość atrybutu `passw`. O ile w przypadku loginu użytkownika widzimy ten sam ciąg znaków, który został wprowadzony w formularzu, to już hasło nie przypomina tego, które wpisaliśmy. Wprawne oko zauważy, że ciąg ten jest dość charakterystyczny i sugeruje kodowanie z wykorzystaniem algorytmu Base64[15]. Możemy się o tym przekonać, kopiując go, a następnie przechodząc do zakładki DECODER.

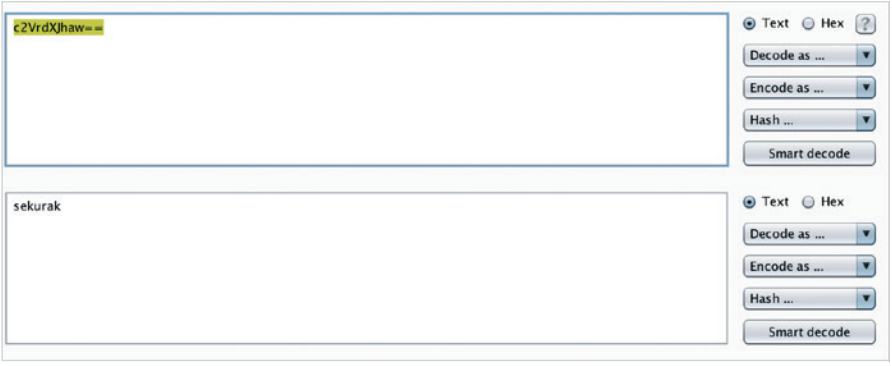
Narzędzie Decoder to kolejne rozszerzenie wbudowane w Burp Suite. Pozwala ono na konwertowanie danych pomiędzy różnymi formatami. Możemy za jego

pomocą obliczyć wynik funkcji skrótu dla wybranego ciągu znaków czy też dokonać konwersji tekstu na odpowiadające poszczególnym znakom kody ASCII.

Zaraz po przejściu do zakładki DECODER ujrzymy widok jak na rysunku 34.

Rysunek 34. Domyślny widok narzędzia Decoder

Wklejmy zatem skopiowany wcześniej ciąg znaków w pole tekstowe, a następnie z listy rozwijanej DECODE AS wybierzmy opcję BASE64. Pojawi się nowe pole tekstowe, które będzie zawierać zdekodowaną wersję hasła (rysunek 35).



Rysunek 35. Zdekodowana postać ciągu znaków

Wygląda na to, że udało nam się zdekodować hasło!

ĆWICZENIE

Przećwicz wykorzystanie pozostałych opcji, jakie udostępnia Decoder. Spróbuj zdekodować zamieszczone poniżej ciągi znaków.

- `73656b7572616b2e706c`
- `c2VrdXJhay5wbA==`
- `&#x72;&#x6f;&#x7a;&#x77;&#x61;&#x6c;&#x2e;&#x74;&#x6f;`
- `%73%65%6b%75%72%61%6b%2e%70%6c`

SEQUENCER – ANALIZA ENTROPII I NIE TYLKO

Testując i analizując poszczególne funkcje aplikacji wykorzystywanej do nauki obsługi Burpa, mieliśmy już okazję kilkukrotnie podejrzeć i przeanalizować treść zapytań wysyłanych przez przeglądarkę do aplikacji, jak i odpowiedzi zwracanych

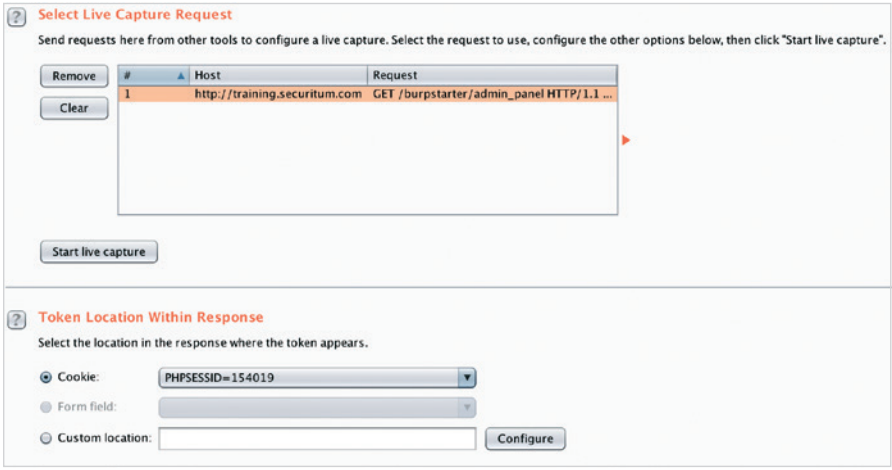
przez serwer. Możliwe, że naszą uwagę przykuł nagłówek `Set-Cookie`, w którym przekazywany jest identyfikator.

Listing 4. Przykładowa treść odpowiedzi HTTP z nagłówkiem Set-Cookie

```
HTTP/1.1 200 OK
Date: Mon, 21 Dec 2020 10:45:41 GMT
Server: Apache
Set-Cookie: PHPSESSID=220056; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Vary: Accept-Encoding
Content-Length: 1650
Connection: close
Content-Type: text/html; charset=UTF-8
```

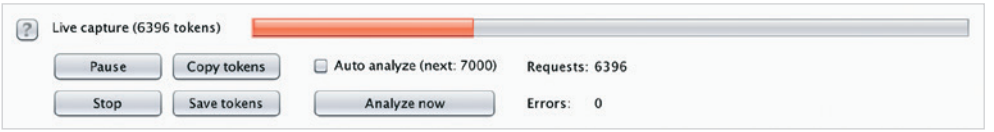
Wartość ciasteczka `PHPSESSID` to wspomniany wcześniej identyfikator sesji. Już na pierwszy rzut oka można stwierdzić, że nie jest on dostatecznie długi – składa się z zaledwie sześciu znaków, a dodatkowo stanowią go jedynie cyfry! Jeżeli zauważymy coś takiego, warto zweryfikować wskaźnik **losowości**, jak określa to dokumentacja[16] samego Burpa. Kombajn do testów bezpieczeństwa, jakim jest Burp, posiada oczywiście narzędzie, dzięki któremu ustalimy poziom entropii identyfikatora sesji – to Sequencer.

Podobnie jak w przypadku narzędzi Repeater, Intruder czy Comparer, Sequencer również musi zostać zasilony w dane. Wybierzmy dowolną odpowiedź z zakładki HTTP HISTORY posiadające nagłówek `Set-Cookie`, a następnie w menu rozwijanym wskażmy opcję SEND TO SEQUENCER. Po wykonaniu tej czynności i przejściu do zakładki SEQUENCER powinien pojawić się widok podobny do tego z rysunku 36.



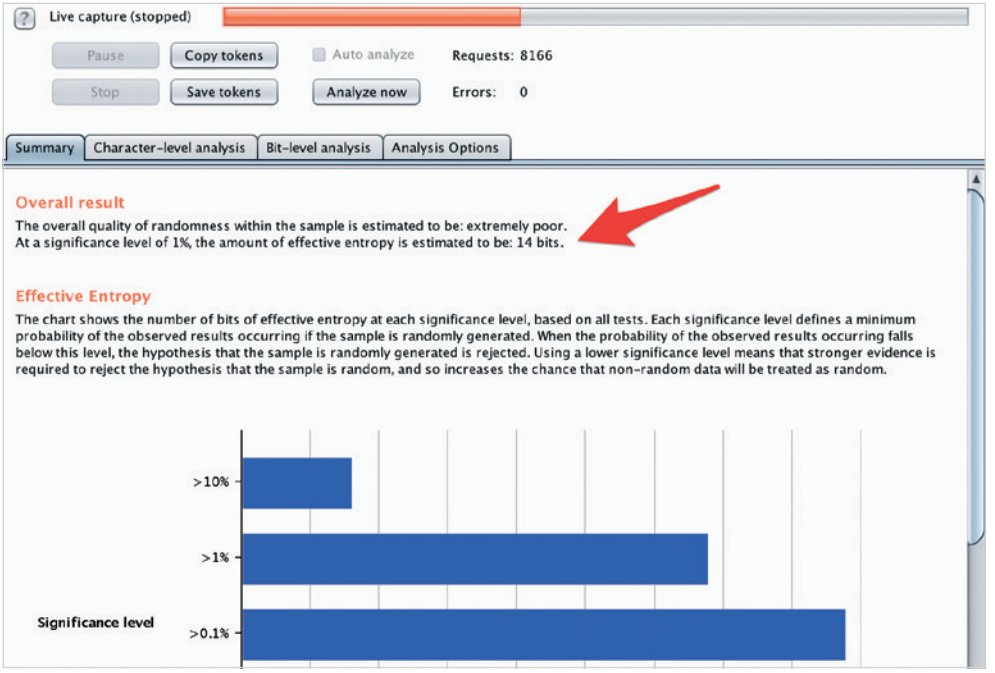
Rysunek 36. Formularz konfiguracji narzędzia Sequencer

Na liście z sekcji SELECT LIVE CAPTURE REQUEST możemy znaleźć wybrane przez nas zapytanie. Możemy też zauważyć, że Sequencer automatycznie wykrył token, który będziemy chcieli poddawać analizie – sekcja TOKEN LOCATION WITHIN RESPONSE. Aby rozpocząć proces analizy, musimy wybrać przycisk START LIVE CAPTURE. Po wybraniu tej opcji w nowym oknie rozpocznie się proces zbierania danych (rysunek 37).



Rysunek 37. Sequencer zbiera dane do analizy

Gdy uznamy, że liczba zebranych próbek jest wystarczająca, możemy zatrzymać proces wysyłania kolejnych zapytań przyciskiem STOP. Aby uruchomić proces analizy, musimy wybrać opcję ANALYZE NOW. Po chwili Sequencer wyświetli w tym samym oknie podsumowanie analizy (rysunek 38). Możemy się z niej dowiedzieć, że entropia (prościej mówiąc: losowość) wykorzystywanego w aplikacji identyfikatora sesji wynosi 14 bitów, co nie stanowi wartości nawet zbliżonej do 64 bitów zalecanych przez organizację OWASP[17].



Rysunek 38. Wynik analizy tokenów

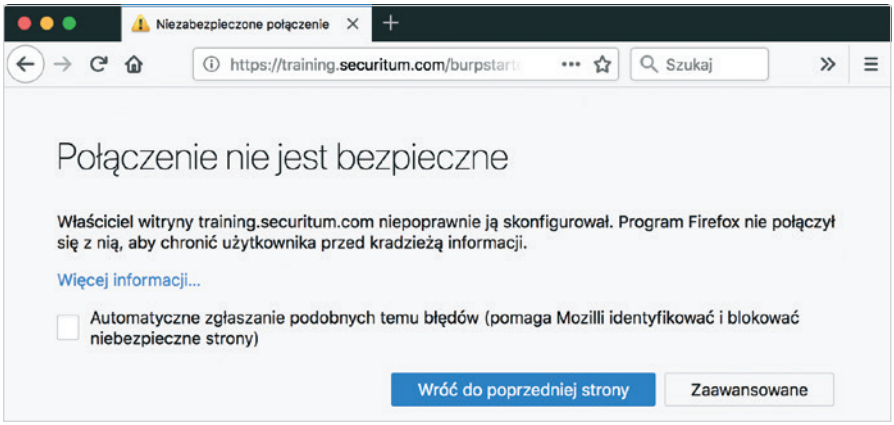
Oszacowanie entropii tokena sesyjnego to nie jedyne, czego można dowiedzieć się z wyników prezentowanych przez Sequencer. Bardziej szczegółowe wyniki możemy znaleźć w zakładkach CHARACTER-LEVEL ANALYSIS oraz BIT-LEVEL ANALYSIS.

ĆWICZENIE

Zweryfikuj, jak liczba zebranych próbek wpływa na jakość wyników. Ile minimalnie trzeba ich zebrać, aby otrzymać wynik podobny do wyżej opisanego?

POŁĄCZENIE NIE JEST BEZPIECZNE – HTTPS

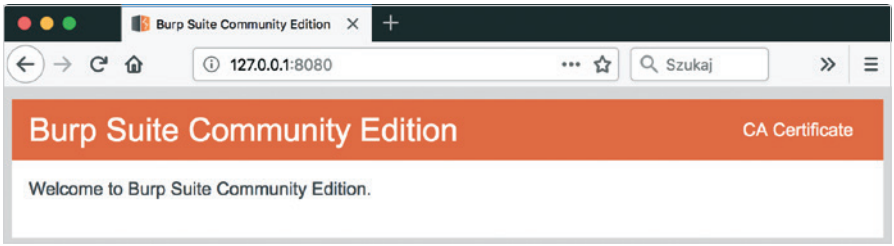
Możliwe, że podczas nauki obsługi Burpa spróbowałeś już przechwycić ruch do innej aplikacji niż ta, którą wykorzystujemy do testów. Z dużym prawdopodobieństwem efekt tego działania był podobny do tego z rysunku 39.



Rysunek 39. Komunikat przeglądarki o próbie nawiązania „niebezpiecznego" połączenia

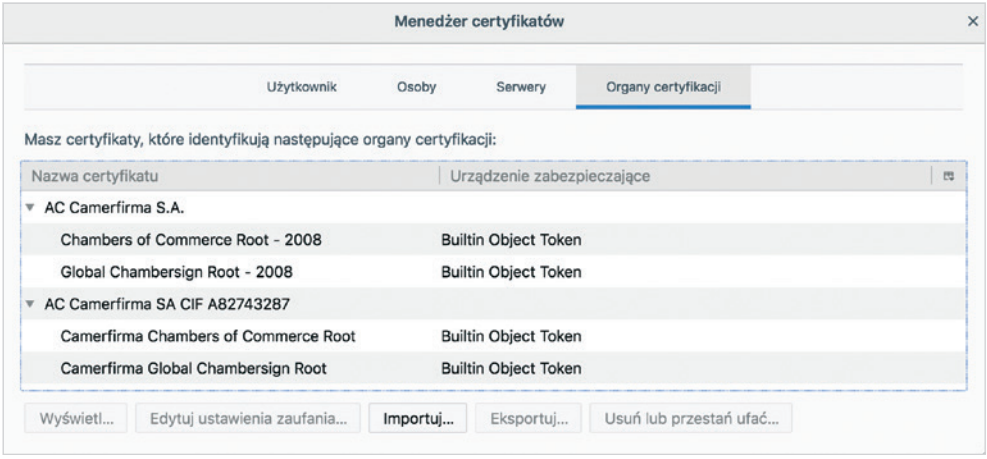
Ostrzeżenie wyświetlone przez przeglądarkę oznacza, że certyfikat, jaki przedstawił serwer *training.securitum.com*, nie należy do listy zaufanych. Inaczej mówiąc, certyfikat, który ma stanowić podstawę do nawiązania bezpiecznego połączenia, nie został poprawnie potwierdzony przez przeglądarkę. W tym miejscu powstaje pytanie: czy i jak możemy testować aplikacje wykorzystujące bezpieczny szyfrowany kanał komunikacji HTTPS?

Istnieje sposób na rozwiązanie tego problemu. W pierwszej kolejności musimy przejść w przeglądarce pod adres *127.0.0.1:8080*. Jest to ten sam adres, pod którym nasłuchuje proxy. Przeglądarka wyświetli stronę podobną do tej z rysunku 40.



Rysunek 40. Zasób udostępniany przez proxy Burp

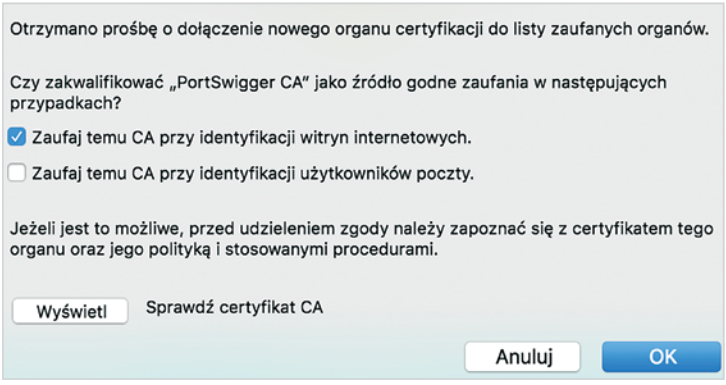
Teraz wybieramy CA CERTIFICATE (prawy górny róg). Odnośnik prowadzi do pliku `cacert.der`, który zawiera certyfikat. Zapiszmy go na dysku, a następnie przejdźmy do ustawień przeglądarki (menu PREFERENCJE / OPCJE). Podobnie jak przy konfiguracji ustawień proxy przeglądarki, wprowadźmy w pole wyszukiwania słowo „certyfikat". Zawęża to dostępne opcje do tych, które związane są z certyfikatami. W kolejnym kroku wybieramy przycisk WYŚWIETL CERTYFIKATY. Przeglądarka wyświetli nowe okno, jak na rysunku 41.



Rysunek 41. Formularz importu certyfikatu

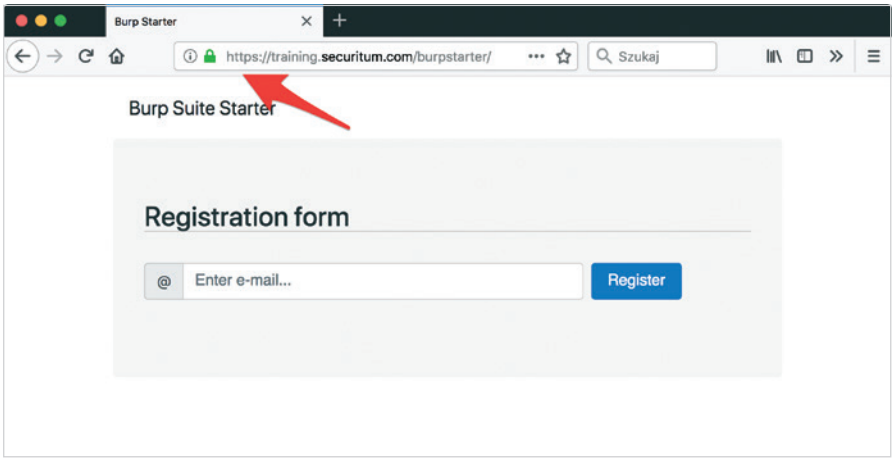
Po przejściu do zakładki ORGANY CERTYFIKACJI musimy wybrać przycisk IMPORTUJ, a następnie w oknie dialogowym wskazać pobrany wcześniej plik certyfikatu.

Po wybraniu pliku przeglądarka wyświetli jeszcze okno z prośbą o potwierdzenie dołączenia nowego certyfikatu. Powinniśmy zaznaczyć opcję ZAUFAJ TEMU CA PRZY IDENTYFIKACJI WITRYN INTERNETOWYCH, a następnie wybrać przycisk OK (rysunek 42).



Rysunek 42. Formularz, w którym potwierdzamy import certyfikatu

Spróbujmy jeszcze raz przejść pod adres *https://training.securitum.com/burpstarter*.



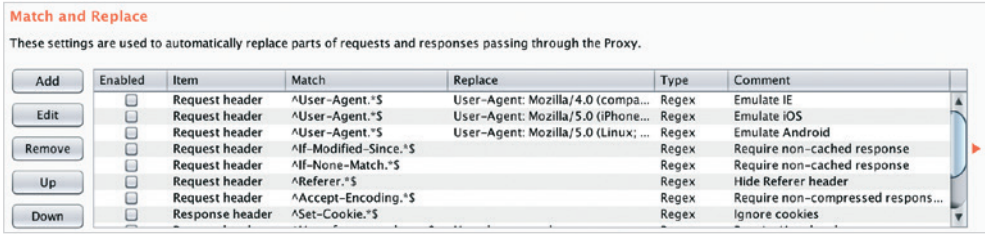
Rysunek 43. Pomyślne przechwycenie zapytania do aplikacji, która wykorzystuje HTTPS

Teraz możemy testować aplikacje, które wykorzystują protokół HTTPS (rysunek 43)!

DOPASUJ I ZAMIEŃ

Może się zdarzyć, że w prowadzonych testach użyteczna okaże się zmiana czegoś w kodzie aplikacji, zanim trafi on do przeglądarki. Podobnie sytuacja może wyglądać w przypadku, gdy będziemy chcieli zmienić coś w zapytaniu wysyłanym do serwera.

Teoretycznie takie zadanie może być zrealizowane przez standardowe przechwytywanie oraz modyfikację zapytań. Jeżeli jednak chcemy wprowadzić jakieś zmiany na stałe, takie podejście będzie niepraktyczne. Ręczna modyfikacja zajmuje mnóstwo czasu i w dłuższej perspektywie będzie wymagała od nas stoickiego wręcz spokoju. Warto więc przejść do zakładki PROXY, a następnie OPTIONS i zapoznać się z opcją MATCH AND REPLACE (rysunek 44), która tego typu zadanie wykona za nas automatycznie.



Rysunek 44. Formularz konfiguracji mechanizmu Match and Replace

Sekcja MATCH AND REPLACE składa się z listy reguł oraz pięciu przycisków sterujących. Spróbujmy dodać regułę, która usunie z naszej testowej aplikacji wymóg wpisania w formularzu rejestracji poprawnego adresu e-mail.

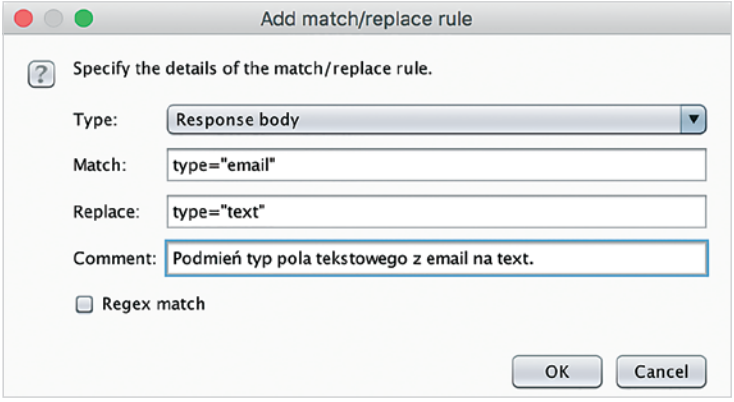
Analizując kod aplikacji, możemy zauważyć, że za wymóg wpisania poprawnego adresu e-mail w polu tekstowym odpowiada atrybut `type` o wartości `email`.

Listing 5. Fragment kodu HTML formularza logowania

```
[...]
<div class="input-group mb-2 mr-sm-2 mb-sm-0">
<div class="input-group-addon">@</div>
<input type="email" autocomplete="off" size=52 name="email" ⤶
class="form-control" placeholder="Enter email...">
</div>
[...]
```

Zastosowanie takiej konstrukcji wymusza na przeglądarce wdrożenie wewnętrznej walidacji. Przeglądarka zaakceptuje tylko te dane, które będą spełniały odpowiednie reguły. Spróbujmy więc tak zmodyfikować treść odpowiedzi, którą przesyła serwer, aby zamiast atrybutu `type` z wartością `email` pojawiła się tam wartość `text`. Jeżeli nam się uda, będzie to oznaczało, że przeglądarka powinna pozwolić na wpisanie w pole `email` dowolnego ciągu znaków.

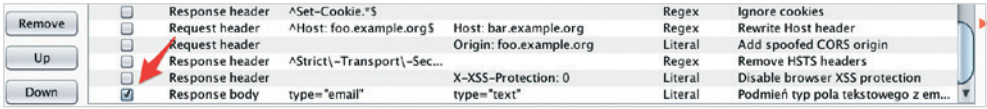
Wróćmy do zakładki OPTIONS, a następnie wybierzmy przycisk ADD z sekcji MATCH AND REPLACE. Burp wyświetli nowe okno, w którym powinniśmy wprowadzić dane i ustawić opcje tak jak na rysunku 45.



Rysunek 45. Konfiguracja nowej reguły Match and Replace

Przeanalizujmy wybrane ustawienia. Pole TYPE zostało ustawione na wartość RESPONSE BODY, co jest zgodne z założeniami. Chcemy i będziemy modyfikować to, co serwer zwraca w odpowiedzi HTTP. Pole MATCH zawiera ciąg znaków, który chcemy znaleźć w odpowiedzi i podmienić. Natomiast pole REPLACE stanowi wartość, która zostanie wstawiona w miejsce ciągu znaków zdefiniowanego w MATCH. Jeżeli wszystko się zgadza, potwierdźmy utworzenie nowej reguły przyciskiem OK.

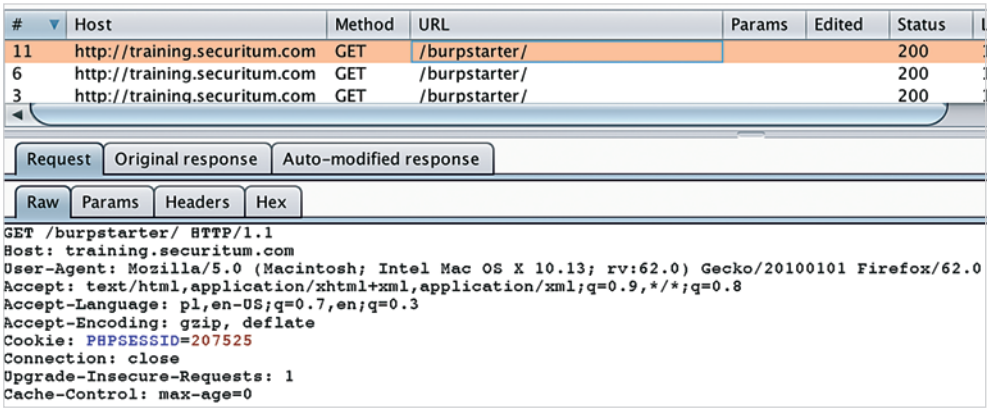
Przewijając listę reguł, powinniśmy znaleźć tę, którą przed chwilą utworzyliśmy (rysunek 46).



Rysunek 46. Nowa reguła widoczna na liście

Musimy jeszcze zweryfikować, czy reguła jest na pewno aktywna. Możemy to zrobić, upewniając się, czy w kolumnie ENABLED znajduje się odpowiedni znak.

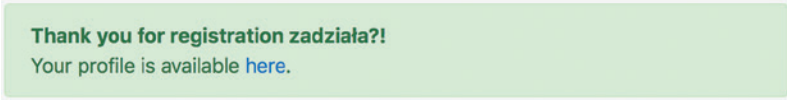
Wróćmy teraz do naszej testowej aplikacji i wybierzmy przycisk ODŚWIEŻ. Gdy to zrobimy, możemy przejść z powrotem do Burpa i znaleźć w zakładce HTTP HISTORY najświeższe zapytanie (rysunek 47).



Rysunek 47. Zakładki, jakie Burp wyświetla w przypadku zmodyfikowania odpowiedzi na skutek działania mechanizmu Match and Replace

Naszą uwagę powinno przykuć to, że zamiast jednej zakładki RESPONSE Burp wyświetla dwie: jedną nazwaną ORIGINAL RESPONSE i drugą – AUTO-MODIFIED RESPONSE. Nazwy zakładek są jednoznaczne. W pierwszej z nich znajdziemy oryginalną treść odpowiedzi zwróconej przez serwer, a w drugiej – wersję po modyfikacji, wyzwoloną ze względu na utworzoną przez nas regułę „dopasuj i zamień".

Przyszła pora na zweryfikowanie, czy wprowadzone przez nas zmiany przyniosły zamierzony skutek. Wprowadźmy w formularz rejestracji dowolny ciąg znaków i sprawdźmy, jak zareaguje przeglądarka.



Rysunek 48. Aplikacja po modyfikacji zaakceptowała losowy ciąg znaków zamiast adresu e-mail

Jak widać na rysunku 48, po wpisaniu w pole tekstowe słowa „zadziała?" przeglądarka nie zwróciła żadnego błędu. Udało nam się zatem skonfigurować Burpa

tak, by automatycznie usuwał walidację, która zaimplementowana jest po stronie przeglądarki.

ĆWICZENIE

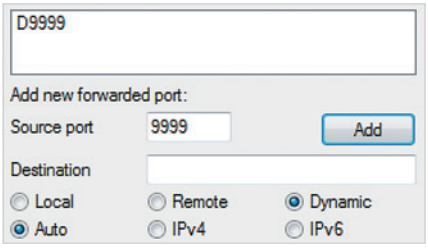
Przygotuj regułę modyfikującą treść zapytania HTTP (*request* HTTP) tak, by część serwerowa aplikacji otrzymała informację o tym, że użytkownik wykorzystuje przeglądarkę Internet Explorer 6.0. Przykładowy ciąg `User-Agent` identyfikujący wersję przeglądarki (Internet Explorer 6.0): `Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)`.

SOCKS PROXY – PROXY W PROXY

Wyobraźmy sobie, że dostęp do aplikacji, którą chcemy testować, możliwy jest tylko z jednego określonego adresu IP. Może się również zdarzyć tak, że aplikacja akceptuje wyłącznie połączenia z adresów IP, dla których mechanizmy geolokalizacji zwracają informację o konkretnym kraju lub kontynencie. W takiej sytuacji musimy mieć dostęp do serwera przesiadkowego, zlokalizowanego w odpowiednim miejscu lub mającego zagraniczny adres IP. Jeżeli już uzyskamy dostęp do takiej maszyny, możemy poinstruować Burpa, by cały ruch sieciowy przekierowywał właśnie przez ten serwer. Nasze proxy będzie wykorzystywało inne proxy do komunikacji ze światem zewnętrznym.

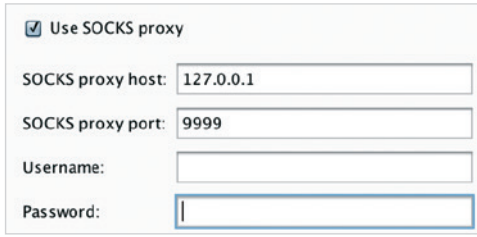
W pierwszym kroku musimy odpowiednio nawiązać połączenie z serwerem pośredniczącym. Jeżeli pracujemy na systemach z rodziny Unix lub Linux, możemy wydać w konsoli polecenie: `ssh -D 9999 uzytkownik@adres_serwera`.

Jeżeli pracujemy na systemie Windows, możemy wykorzystać oprogramowanie PLINK lub PuTTY[18]. Składnia dla PLINK będzie identyczna jak dla klienta SSH z platformy Unix/Linux. Wystarczy, że w konsoli systemu uruchomimy pobrany plik `plink` z takimi samymi parametrami. W przypadku programu PuTTY musimy przejść do zakładki TUNNELS, w pole SOURCE PORT wpisać wartość `9999` oraz zaznaczyć opcję DYNAMIC. Zmiany zatwierdzamy poprzez wybranie przycisku ADD (rysunek 49). Następnie możemy wrócić do zakładki SESSION, wprowadzić adres serwera w pole HOST NAME i wybrać przycisk OPEN. Po chwili powinna nastąpić próba połączenia do serwera.



Rysunek 49. Konfiguracja przekazywania portów w PuTTY

Gdy już nawiążemy połączenie z serwerem, musimy jeszcze skonfigurować Burpa tak, aby przekierowywał ruch na odpowiedni port. W tym celu powinniśmy przejść do zakładki USER OPTIONS i w sekcji SOCKS PROXY wprowadzić dane jak na rysunku 50. Uwaga: zaznaczenie opcji USE SOCKS PROXY będzie możliwe dopiero po uzupełnieniu pól SOCKS PROXY HOST oraz SOCKS PROXY PORT.



Rysunek 50. Konfiguracja Burpa do pracy z użyciem SOCKS PROXY

Przechodząc teraz do jednego z portali, np. *https://www.whatismyip.com/* lub *http://ifconfig.io/*, będziemy mogli potwierdzić, że adres IP, z którego komunikujemy się z siecią Internet, to adres naszego serwera przesiadkowego. Burp przekazuje i odbiera ruch właśnie z tego serwera!

Jeżeli chcemy zaprzestać wykorzystania serwera pośredniczącego, powinniśmy dezaktywować opcję USE SOCKS PROXY.

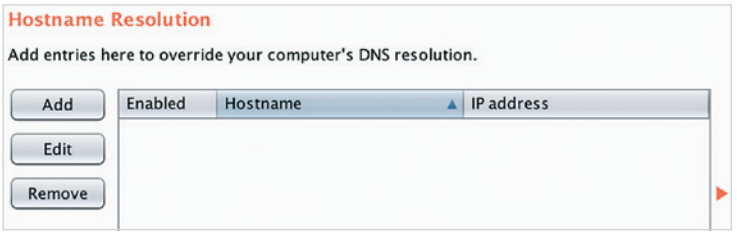
ROZWIĄZYWANIE NAZW I BRAK UPRAWNIEŃ

Może się zdarzyć, że do testów zostanie nam dostarczona aplikacja, dla której nie został w poprawny sposób skonfigurowany serwer DNS. Oznacza to, że wpisując w pasku przeglądarki adres domenowy tej aplikacji, nie będziemy mogli się z nią połączyć. W takim przypadku najprawdopodobniej od opiekunów aplikacji uzyskamy nie tylko adres, ale również IP serwera, na którym zainstalowana jest aplikacja. Bardzo często takiej sytuacji towarzyszy prośba od deweloperów, by „dodać wpis do hostów". Oznacza to, że powinniśmy znaleźć plik `hosts` na dysku naszego komputera, a następnie dodać wpis w formacie podobnym do tego:

```
<adres IP><spacja><nazwa domenowa>
1.2.3.4 sekurak.pl
```

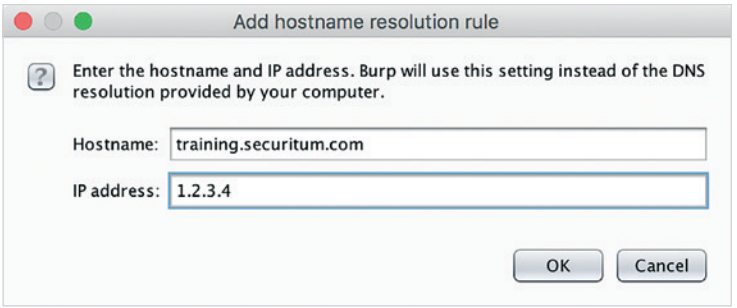
Jedyny problem, z jakim możemy się tutaj spotkać, to brak uprawnień do edycji tego pliku. Zarówno w systemach z rodziny Windows, jak i Unix/Linux potrzebne są do tego uprawnienia administratora. Co, jeżeli takowych nie posiadamy? Z pomocą może tutaj przyjść Burp, którego możemy poinstruować, na jakie adresy IP ma rozwiązywać wskazane nazwy domenowe.

Aby skorzystać z tej opcji, powinniśmy przejść do zakładki PROJECT OPTIONS, następnie CONNECTIONS. W sekcji HOSTNAME RESOLUTION znajdziemy formularz podobny do tego z rysunku 51.



Rysunek 51. Konfiguracja rozwiązywania nazw w Burp Suite

Aby dodać nową regułę rozwiązywania nazw, wybierzmy przycisk ADD. W oknie wprowadźmy najpierw nazwę domeny, a później adres IP, na jaki ma być rozwiązana (rysunek 52). Dodanie reguły zatwierdzamy przyciskiem OK.



Rysunek 52. Konfiguracja adresu IP dla nazwy domenowej

Na wcześniej pustej liście reguł pojawi się nowa pozycja, która domyślnie będzie aktywna. Możemy zweryfikować jej działanie, przechodząc z powrotem do testowej aplikacji lub wybierając przycisk ODŚWIEŻ w oknie przeglądarki. Po chwili przeglądarka wyświetli komunikat z informacją o błędzie połączenia. Poprawność rozwiązania nazwy możemy zweryfikować, przechodząc do zakładki HTTP HISTORY. Jako najświeższy powinniśmy znaleźć wpis podobny do tego z rysunku 53. Jeżeli wszystko poszło zgodnie z założeniami, w kolumnie IP znajdziemy adres, który wprowadziliśmy przy dodawaniu nowej reguły.

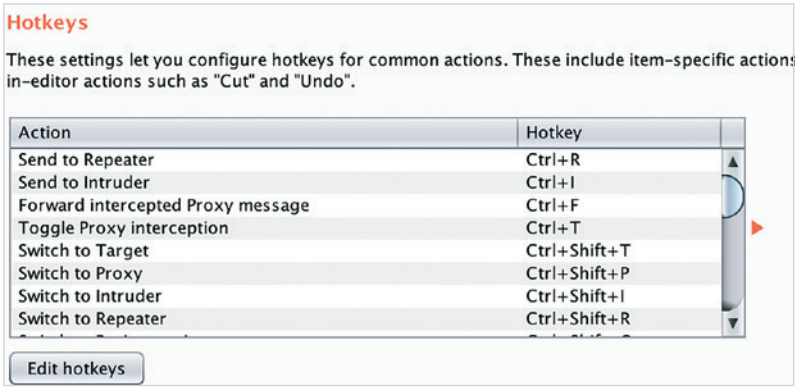
Host	Method	URL	IP
http://training.securitum.com	GET	/	1.2.3.4

Rysunek 53. Burp rozwiązał nazwę training.securitum.com na skonfigurowany adres IP

SKRÓTY KLAWISZOWE

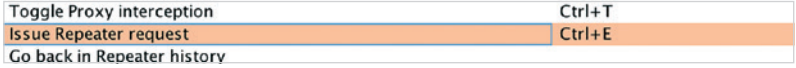
Podczas pracy z Burpem wiele czynności będziemy wykonywali dziesiątki, jeżeli nie setki razy. Skoro więc wybraną akcję możemy wyzwolić skrótem klawiszowym zamiast kliknięciem myszki, to jak najbardziej powinniśmy skorzystać z takiej opcji.

Burp posiada zestaw wbudowanych skrótów klawiszowych, ale pozwala również zmodyfikować domyślne kombinacje, jak i dodawać całkiem nowe reguły. Aby zapoznać się z domyślną listą skrótów, powinniśmy przejść do zakładki USER OPTIONS, a dalej MISC. W sekcji HOTKEYS znajdziemy listę akcji, jakie można wyzwolić w Burpie, oraz przypisane do nich skróty (rysunek 54).



Rysunek 54. Domyślny zestaw skrótów wbudowany w Burp Suite

Wszystkie opcje możemy dopasować według własnego uznania, wybierając przycisk EDIT HOTKEYS. Burp wyświetli nowe okno, w którym po zaznaczeniu wybranej pozycji, a następnie wciśnięciu określonego przez nas skrótu na klawiaturze skrót ten zostanie przypisany do wybranej akcji. Przećwiczmy to na przykładzie akcji ISSUE REPEATER REQUEST. Zaznaczamy tę pozycję na liście, następnie na klawiaturze wciskamy klawisze CTRL oraz E. Efekt powinien być podobny do tego z rysunku 55. Nowe ustawienia zatwierdzamy przyciskiem OK. Od teraz, modyfikując zapytania w narzędziu Repeater, nie będziemy musieli każdorazowo klikać w przycisk SEND – wystarczy, że na klawiaturze wybierzemy ustawiony skrót klawiszowy.



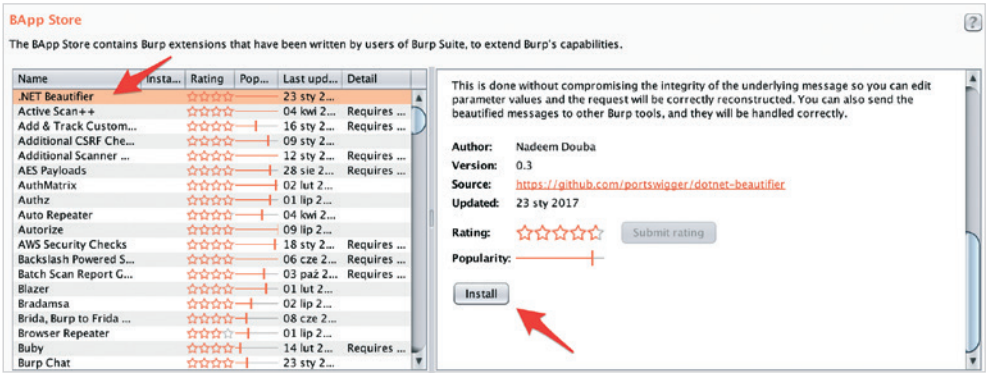
Rysunek 55. Nowy skrót klawiszowy przypisany do akcji

WTYCZKI – JESZCZE WIĘCEJ MOŻLIWOŚCI!

Sprzęt lub oprogramowanie posiadające ogromną liczbę funkcji zwykło się nazywać „kombajnem". Biorąc pod uwagę ilość zastosowań Burpa, zdecydowanie uprawnione jest użycie takiego określenia. Okazuje się jednak, że listę funkcji możemy rozbudowywać poprzez instalację wtyczek lub tworzenie własnych[19] rozszerzeń.

Przechodząc do zakładki EXTENDER, a następnie BAPP STORE, znajdziemy listę gotowych do zainstalowania rozszerzeń. Polecam zapoznać się z opisem każdego z nich – może akurat któraś z tych wtyczek będzie rozszerzać możliwości Burpa o funkcję, która jest nam niezbędna.

Proces instalacji wtyczki odbywa się poprzez wybranie jej na liście, a następnie kliknięcie w przycisk INSTALL na samym dole opisu wtyczki (rysunek 56).



Rysunek 56. Instalacja wtyczki

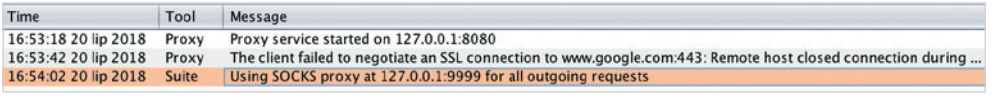
Istnieją tutaj jednak pewne ograniczenia oraz specyficzne wymagania. Niektóre z wtyczek można instalować tylko w pełnej, płatnej wersji Burp Suite. Informację na ten temat znajdziemy w kolumnie DETAIL dla danej wtyczki.

Niektóre z wtyczek, które stworzone zostały z wykorzystaniem języka Python lub Ruby, będą wymagały pobrania i zainstalowania na naszej maszynie odpowiedniego oprogramowania. Dla wtyczek napisanych w Pythonie będzie to Jython[20], a dla Ruby – JRuby[21].

Wiele ciekawych rozszerzeń można znaleźć również na GitHub[22], jednak tutaj należy zachować ostrożność i stosować zasadę ograniczonego zaufania. Wtyczki, które pobieramy z BAPP STORE, przechodzą proces weryfikacji. Odpowiedzialność za to, co instalujemy z innych źródeł, ponosimy sami.

GDY COŚ PRZEBIEGA NIEZGODNIE Z PLANEM

Na koniec przygody z poznawaniem Burp Suite warto jeszcze wspomnieć o narzędziu, które pozwoli zdiagnozować ewentualne problemy, jakie mogą wystąpić podczas testów różnych aplikacji. Mowa tutaj o zakładce ALERTS, w której Burp udostępnia logi (rysunek 57). Jeżeli coś nie idzie po naszej myśli, zawsze warto tam zajrzeć. Istnieje duże prawdopodobieństwo, że jeden z komunikatów zasugeruje nam rozwiązanie problemu.



Rysunek 57. Komunikaty odkładane w zakładce ALERTS

PODSUMOWANIE

Rozpoczęcie przygody z testami aplikacji WWW wymaga od nas poznania narzędzi, które będą nieodłączną częścią tej pracy. Niezwykle ważne jest, by wybrać odpowiednie, dzięki czemu praca będzie przyjemna i efektywna. Wydaje się, że takim narzędziem może być Burp, który domyślnie posiada niemal wszystkie niezbędne funkcje, jakie mogą nam być potrzebne do realizacji testów bezpieczeństwa. Zapoznanie się z jego obsługą należy traktować jako obowiązkowy przystanek, na którym musimy się zatrzymać i spędzić trochę czasu, aby przygotować się do roli testera bezpieczeństwa.

1 *Hypertext Transfer Protocol* [w:] *Wikipedia, wolna encyklopedia*, https://pl.wikipedia.org/wiki/Hypertext_Transfer_Protocol
2 *HTML* [w:] *Wikipedia, wolna encyklopedia*, https://pl.wikipedia.org/wiki/HTML
3 *JavaScript* [w:] *Wikipedia, wolna encyklopedia*, https://pl.wikipedia.org/wiki/JavaScript
4 *Kaskadowe arkusze stylów* [w:] *Wikipedia, wolna encyklopedia*, https://pl.wikipedia.org/wiki/Kaskadowe_arkusze_styl%C3%B3w
5 PortSwigger, *Burp Suite Editions*, https://portswigger.net/burp
6 Telerik, *Fiddler*, https://www.telerik.com/fiddler
7 OWASP, *ZAP (zaproxy)*, https://github.com/zaproxy/zaproxy
8 PortSwigger, *Burp Suite Community Edition v2.1.02*, https://portswigger.net/burp/communitydownload
9 Oracle, *Java SE Downloads*, https://www.oracle.com/technetwork/java/javase/downloads/index.html
10 *VirusTotal*, https://www.virustotal.com/old-browsers/
11 Securitum, *Burp Suite Starter*, http://training.securitum.com/burpstarter/
12 *RFC 3986*, https://tools.ietf.org/html/rfc3986
13 *Percent-encoding* [w:] *Wikipedia, the free encyclopedia*, https://en.wikipedia.org/wiki/Percent-encoding
14 Kali Linux, *dirbuster*, https://gitlab.com/kalilinux/packages/dirbuster/
15 *Base64* [w:] *Wikipedia, wolna encyklopedia*, https://pl.wikipedia.org/wiki/Base64
16 PortSwigger, *Burp Sequencer*, https://portswigger.net/burp/documentation/desktop/tools/sequencer
17 OWASP, *Session Management Cheat Sheet*, https://www.owasp.org/index.php/Session_Management_Cheat_Sheet
18 *Download PuTTY: latest release (0.72)*, https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html
19 PortSwigger, *Writing your first Burp Suite extension*, https://portswigger.net/burp/extender/writing-your-first-burp-suite-extension
20 *Jython: Python for the Java Platform*, https://www.jython.org/download.html
21 *JRuby*, https://www.jruby.org/
22 https://www.google.pl/search?q=site%3Agithub.com+%22burp+suite+extension%22