Bezpieczeństwo przeglądarki internetowej

Duża odpowiedzialność spoczywa na szerokich ramionach skromnej przeglądarki internetowej. Przeglądarka internetowa została zaprojektowana tak, aby żądać instrukcji z całego Internetu, a następnie instrukcje te są wykonywane niemal bez żadnych wątpliwości. Przeglądarka musi wiernie łączyć zdalnie pobraną treść w ustandaryzowaną, przyswajalną formę i obsługiwać bogaty zestaw funkcji dostępnych w dzisiejszym Web 2.0. Pamiętaj, że jest to to samo oprogramowanie, za pomocą którego prowadzisz ważne sprawy - od utrzymania sieci społecznościowych po bankowość internetową. Oczekuje się, że to oprogramowanie będzie Cię chronić, nawet jeśli zapuszczasz się w wiele symbolicznych ciemnych zaułków Internetu. Oczekuje się, że ułatwi wyjście z takiej alejki przy jednoczesnym bezpiecznym zakupie w innej zakładce lub oknie. Wielu zakłada, że ich przeglądarka jest jak samochód pancerny, zapewniając bezpieczne i wygodne środowisko do obserwowania świata zewnętrznego, chroniąc wszystkie aspekty osobistych interesów i odrzucając wszystko, co niebezpieczne. Pod koniec będziesz miał informacje, które pozwolą ci zdecydować, czy jest to rozsądne założenie. Zespół programistów tego oprogramowania musi upewnić się, że każdy z jego licznych zakamarków nie stanowi drogi dla hakera. Niezależnie od tego, czy świadomie o tym wiesz, za każdym razem, gdy używasz przeglądarki, ufasz zespołowi ludzi, których prawdopodobnie nigdy nie spotkałeś (i prawdopodobnie nigdy nie spotkasz), aby chronić Twoje ważne informacje przed atakującymi w Internecie. Ta Część przedstawia metodologię hakowania przeglądarek internetowych, którą można wykorzystać do obraźliwych działań. Poznajesz rolę przeglądarki internetowej w ekosystemie sieciowym, w tym zagłębiasz się w interakcje między nią a serwerem sieciowym. Zapoznasz się również z niektórymi podstawami bezpieczeństwa przeglądarek, które będą stanowić podstawę dla pozostałych Części.

Główna zasada

Zachęcamy do zapomnienia na chwilę o przeglądarce internetowej i zastanowienia się nad czystym płótnem bezpieczeństwa. Wyobraź sobie, że znajdujesz się w takiej sytuacji: jesteś odpowiedzialny za utrzymanie bezpieczeństwa organizacji i musisz podjąć decyzję. Czy wdrażasz oprogramowanie w oparciu o poziom ryzyka, jakie będzie stwarzać? Oprogramowanie zostanie zainstalowane w standardowym środowisku operacyjnym (SOE) dla prawie każdej maszyny w organizacji. Będzie służył do uzyskiwania dostępu do najbardziej wrażliwych danych i przeprowadzania najbardziej wrażliwych operacji. To oprogramowanie będzie podstawowym narzędziem praktycznie dla wszystkich pracowników, w tym dyrektora generalnego, zarządu, administratorów systemu, finansów, zasobów ludzkich, a nawet klientów. Przy całej tej kontroli i dostępie do kluczowych danych biznesowych, z pewnością brzmi to jak wymarzony cel hakera i propozycja wysokiego ryzyka. Ogólne specyfikacje oprogramowania są następujące:

■ Poprosi o instrukcje z Internetu i wykona je.

■ Obrońca nie będzie sprawował kontroli nad tymi instrukcjami.

■ Niektóre instrukcje informują oprogramowanie, aby uzyskać więcej instrukcji od:

- Innego miejsca w Internecie

- Innego miejsca w intranecie

- Niestandardowe porty HTTP i HTTPS TCP

■ Niektóre instrukcje informują oprogramowanie, aby przesyłało dane przez TCP. Może to spowodować ataki na inne urządzenia sieciowe.

■ Będzie szyfrować komunikację z dowolnymi lokalizacjami w Internecie. Obrońca nie będzie mógł zobaczyć komunikacji.

■ Będzie stale zwiększać liczbę ataków, które mogą być celem ataku. Zaktualizuje się w tle bez powiadomienia.

■ Często zależy to od wtyczek, aby umożliwić efektywne wykorzystanie. Nie ma scentralizowanej metody aktualizacji wtyczek.

Ponadto badania terenowe nad oprogramowaniem ujawniają:

■ Wtyczki są ogólnie uważane za mniej bezpieczne niż samo oprogramowanie podstawowe.

■ Każdy wariant oprogramowania ma historię udokumentowanych luk w zabezpieczeniach.

■ Raport analizy bezpieczeństwa1, który podsumowuje ataki na to oprogramowanie jako największe zagrożenie dla przedsiębiorstwa.

Bez wątpienia zorientowałeś się, że odwołujemy się do przeglądarki internetowej. Zapominając raz jeszcze o tym i wydarzeniach z historii i wracając do naszego pustego płótna bezpieczeństwa, szaleństwem byłoby nie kwestionować sensowności wdrażania tego oprogramowania. Nawet bez korzystania z danych z terenu jego specyfikacje wydają się niezwykle niepokojące z punktu widzenia bezpieczeństwa. Jednak cała ta dyskusja jest oczywiście czysto koncepcyjna w prawdziwym świecie. Przeszliśmy już daleko od punktu, z którego nie ma powrotu, a biorąc pod uwagę masę krytyczną witryn, nikt nie może zadecydować, że przeglądarka internetowa stanowi potencjalnie poważne zagrożenie bezpieczeństwa i jako taka nie zostanie udostępniona każdemu członkowi personelu. Jak już wiesz, wdrażane są dosłownie miliardy przeglądarek internetowych. Nie udostępnienie przeglądarki internetowej pracownikom organizacji prawie na pewno wpłynie negatywnie na ich produktywność. Nie wspominając o tym, że byłoby to uważane za środek raczej drakoński lub zacofany. Przeglądarka internetowa ma coraz więcej zastosowań i przedstawia różne wyzwania związane z hakowaniem i bezpieczeństwem w zależności od kontekstu użytkowania. Przeglądarka jest tak wszechobecna, że wiele osób nietechnicznych postrzega ją jako „Internet”. Mają ograniczoną ekspozycję na inne przejawy danych, które może wywołać protokół internetowy. W dobie Internetu daje to przeglądarce niezaprzeczalnie dominującą pozycję w życiu codziennym, dlatego też branża informatyczna jest do niej przywiązana. Przeglądarka internetowa jest prawie wszędzie w sieci - w strefie sieci użytkownika, strefach gości, a nawet bezpiecznych strefach DMZ. Nie zapominaj, że w wielu przypadkach administratorzy użytkowników muszą zarządzać swoimi urządzeniami sieciowymi za pomocą przeglądarek internetowych. Producenci wskoczyli na modę internetową i wykorzystali dostępność przeglądarek, zamiast odkrywać koło na nowo. Poleganie na tym oprogramowaniu do przeglądania stron internetowych jest niczym absolutnym. W dzisiejszym świecie efektywniej jest pytać, gdzie w sieci nie ma przeglądarki, niż gdzie się znajduje.

Eksploracja przeglądarki

Kiedy dotkniesz internetu, internet od razu dotyka Ciebie. W rzeczywistości, niezależnie od tego, czy świadomie to sobie uświadamiasz, czy nie, zapraszasz go, by się z tobą skontaktował. Prosisz go o skorzystanie z różnych środków bezpieczeństwa wprowadzonych w celu ochrony Twojej sieci i wykonanie instrukcji, nad którymi masz tylko kontrolę na wysokim poziomie, a wszystko to w imię renderowania strony i dostarczania na Twój ekran nieznanej/niezaufanej treści. Przeglądarka działa z zestawem uprawnień nadawanych jej przez system operacyjny, identycznymi jak każdy inny program w przestrzeni użytkownika. Te uprawnienia są równoważne z tymi, które przydzielono Tobie, użytkownikowi! Nie zapominajmy, że dane wprowadzane przez użytkownika są zawsze niczym więcej

niż zestawem instrukcji do aktualnie uruchomionego programu - nawet jeśli ten program jest Eksploratorem Windows lub powłoką UNIX. Jedyną różnicą między danymi wejściowymi użytkownika a danymi otrzymanymi z dowolnego innego źródła jest zróżnicowanie narzucone przez program otrzymujący dane wejściowe! Kiedy zastosujesz to zrozumienie do przeglądarki internetowej, której podstawową funkcją jest odbieranie i wykonywanie instrukcji z dowolnych miejsc w świecie zewnętrznym, potencjalne zagrożenia z tym związane stają się bardziej oczywiste.

Symbioza z aplikacją internetową

Sieć wykorzystuje szeroko rozpowszechnione podejście sieciowe zwane modelem klient-serwer, które zostało opracowane w latach 70. XX wieku. Komunikuje się za pomocą procesu requestresponse, w którym przeglądarka internetowa realizuje żądanie, a serwer sieciowy odpowiada odpowiedzią. Ani serwer sieciowy, ani klient sieciowy nie mogą tak naprawdę wykorzystać swojego potencjału bez drugiego. Są prawie całkowicie współzależni; przeglądarka internetowa nie miałaby prawie nic do wyświetlenia, a serwer sieciowy nie miałby żadnego celu w udostępnianiu swojej zawartości. Ta zasadnicza symbioza tworzy niezliczone dynamiczne splecione pasma sieci. Więź między tymi dwoma kluczowymi elementami rozciąga się również na postawę bezpieczeństwa. Bezpieczeństwo przeglądarki internetowej może wpływać na aplikację internetową i odwrotnie. Niektóre kontrolki można zabezpieczyć oddzielnie, ale wiele z nich zależy od ich odpowiedników. W wielu przypadkach to związek między przeglądarką a aplikacją należy wzmocnić lub, z perspektywy hakera, zaatakować. Na przykład, gdy serwer sieciowy ustawia plik cookie na określone pochodzenie, oczekuje się, że przeglądarka internetowa będzie respektować tę dyrektywę i nie udostępniać (potencjalnie wrażliwego) pliku cookie innym źródłom. Bezpieczeństwo zaangażowania przeglądarki internetowej w aplikację internetową należy rozumieć w kontekście. W wielu przypadkach dyskusje będą zagłębiać się w interakcje między tymi dwoma komponentami. Wykorzystywanie relacji między tymi dwoma podmiotami omówiono w kolejnych sekcjach. Zachęcamy do dalszych badań nad lukami w aplikacjach internetowych.

Zasady tego samego pochodzenia

Najważniejszą kontrolą bezpieczeństwa w przeglądarce internetowej jest Polityka Same Origin, znana również jako SOP. Ta kontrola ogranicza zasoby z jednego źródła współdziałające z innymi źródłami. SOP uważa, że strony mające tę samą nazwę hosta, schemat i port mają ten sam początek. Jeśli którykolwiek z tych trzech atrybutów jest inny, zasób ma inne pochodzenie. Dlatego pod warunkiem, że zasoby pochodzą z tej samej nazwy hosta, schematu i portu, mogą współdziałać bez ograniczeń. SOP początkowo została zdefiniowana tylko dla zasobów zewnętrznych, ale została rozszerzona o inne rodzaje źródeł. Obejmuje to dostęp do plików lokalnych przy użyciu schematu file:// i zasobów związanych z przeglądarką przy użyciu schematu chrome://. Wiele innych schematów jest obsługiwanych przez dzisiejsze przeglądarki.

Nagłówki http

Możesz myśleć o nagłówkach HTTP jako o adresie i innych instrukcjach zapisanych na kopercie, które dyktują, gdzie pakiet powinien się znaleźć i jak powinna być obsługiwana zawartość pakietu. Przykładami mogą być „Fragile: Obchodź się ostrożnie”, „Trzymaj się płasko” lub „Niebezpieczeństwo: materiały wybuchowe!” Są to główne dyrektywy, których używa protokół HTTP do dyktowania, co należy zrobić z następującą zawartością. Klienci sieci Web dostarczają nagłówki HTTP na początku wszystkich żądań do serwera sieci Web, a serwery sieci Web odpowiadają nagłówkami HTTP jako pierwszym elementem każdej odpowiedzi. Treść nagłówków określa, w jaki sposób następująca treść jest przetwarzana przez serwer sieciowy lub przeglądarkę internetową. Niektóre nagłówki są

wymagane, aby interakcja mogła działać; inne są opcjonalne, a niektóre mogą być używane wyłącznie w celach informacyjnych.

Języki znaczników

Języki znaczników to sposób na określenie sposobu wyświetlania treści. W szczególności definiują one znormalizowany sposób tworzenia symboli zastępczych dla danych i symboli zastępczych dla adnotacji związanych z danymi w tym samym dokumencie. Każda strona internetowa, którą widziałeś w swoim życiu, prawdopodobnie używała języka znaczników, aby przekazać przeglądarce internetowej instrukcje dotyczące wyświetlania strony. Istnieją różne rodzaje języków znaczników. Niektóre języki znaczników są bardziej popularne niż inne, a każdy ma swoje mocne i słabe strony. Jak zapewne już wiesz, HTML jest preferowanym językiem znaczników przeglądarki internetowej.

HTML

HyperText Markup Language lub HTML to podstawowy język programowy używany do wyświetlania stron internetowych. Chociaż początkowo rozszerzono go ze standardowego uogólnionego języka znaczników (SGML), obecny HTML przeszedł wiele zmian od tego czasu. Bezwzględna zależność od znaczników (współistnienie danych i adnotacji lub instrukcji) jest podstawową przyczyną kilku ważnych, trwałych i systemowych problemów związanych z bezpieczeństwem. Dowiesz się więcej o HTML

XML

XML jest bliskim związkiem z HTML. Jeśli znasz HTML, XML nie będzie dla Ciebie zbyt wielkim wyzwaniem. Chociaż żadne z nich nie jest szczególnie przyjemny dla ludzkiego oka, oba zapewniają bardzo bogaty sposób przedstawiania złożonych danych. Często spotykasz się z XML w sieci Web, zwykle jako transport dla usług sieci Web lub inne interakcje zdalnego wywoływania procedur (RPC).

Kaskadowe arkusze stylów

Kaskadowe arkusze stylów (CSS) to główna metoda używana przez przeglądarki internetowe do określania stylu zawartości strony internetowej (nie mylić z XSS, który jest akronimem luki bezpieczeństwa Cross-site Scripting). CSS umożliwia oddzielenie treści od jej stylu. Bardzo podstawowym tego przykładem jest pogrubienie zdania. Oczywiście CSS jest znacznie potężniejszy niż ten prosty przykład i rozciąga się na złożoność obserwowaną w sieci.

Skrypty

Języki skryptów internetowych to sztuka, której warto się nauczyć! Jeśli wchodzisz w interakcję z siecią na poziomie technicznym, w pewnym momencie natkniesz się na nie. Ogólnie rzecz biorąc, tworzenie skryptów jest warunkiem wstępnym do pracy w informatyce, która w jakiś sposób wkradła się i zajęła bardzo widoczną pozycję w przeglądarce. Z kolejnych rozdziałów dowiesz się, że hakerzy wykorzystują skrypty w przeglądarce do uruchamiania niektórych z najczęstszych exploitów, w tym XSS. Ta wiedza będzie ci potrzebna w swoim arsenale.

JavaScript

JavaScript obsługuje koncepcje programowania funkcjonalnego i obiektowego. W przeciwieństwie do Javy, która jest językiem silnie typizowanym, JavaScript jest typowany luźno. Język ma dominującą pozycję w ekosystemie internetowym i będzie dostępny w najbliższej przyszłości. Domyślnie działa w każdej przeglądarce. Zrozumienie JavaScript jest niezbędne dla czytelnika, ponieważ większość kodu w tej książce używa tego języka. Ataki napisane w JavaScript (bez względu na dziwactwa przeglądarki) są

kompatybilne we wszystkich przeglądarkach. To sprawia, że jest to fantastyczny język bazowy do hakowania przeglądarek.

VBScript

VBScript jest obsługiwany tylko w przeglądarkach Microsoft i jest rzadko używany w poważnym tworzeniu stron internetowych. Dzieje się tak, ponieważ nie obsługuje wielu przeglądarek. Jest to alternatywa Microsoftu dla JavaScript, a jej pochodzenie sięga wczesnych wojen przeglądarek. Wiele jego funkcji można osiągnąć w JavaScript. Oczywiście rodzi to pytanie, czy VBScript jest w ogóle potrzebny. Jeśli już, wydaje się, że jest to powrót do czasów, kiedy Internet Explorer miał całkowitą dominację w tej przestrzeni.

Obiektowy model dokumentu

Obiektowy model dokumentu (powszechniej określany jako DOM) jest podstawową koncepcją przeglądarki internetowej. Jest to API do interakcji z obiektami w dokumentach HTML lub XML. DOM zapewnia metodę interakcji języków skryptowych z silnikiem renderującym poprzez dostarczanie odwołań do elementów HTML w postaci obiektów. DOM jest efektywnie połączony z JavaScript (lub innymi wybranymi językami skryptowymi). Został stworzony, aby umożliwić zdefiniowaną metodę dostępu do żywego wyrenderowanego dokumentu, tak aby skrypty działające w przeglądarce mogły odczytywać i/lub pisać do niego dynamicznie. Pozwoliło to na zmianę strony bez wysyłania nowych żądań do serwera WWW i bez konieczności interakcji użytkownika.

Silniki renderujące

Silniki renderujące są nazywane różnymi nazwami w kontekście przeglądarek internetowych, włączając w to mechanizmy układu graficznego i silniki przeglądarek internetowych.5 Nazwy te są używane zamiennie w całej książce. Te komponenty odgrywają kluczową rolę w ekosystemie przeglądarek. Odpowiadają za konwersję danych do formatu przydatnego do prezentacji użytkownikowi na ekranie. Przeglądarka internetowa prawdopodobnie użyje HTML i obrazów w połączeniu z CSS, aby stworzyć końcowy produkt graficzny, który użytkownicy widzą w swojej przeglądarce internetowej. To właśnie te silniki zapewniają użytkownikowi wrażenia graficzne. Chociaż zwykle określa się je w sensie graficznym, istnieją silniki renderujące tekst również takie jak Lynx i W3M. W sieci używa się wielu silników renderujących. Popularne graficzne silniki renderujące omówione w tej książce to WebKit, Blink, Trident i Gecko.

WebKit

WebKit jest najpopularniejszym silnikiem renderującym i jest używany w wielu przeglądarkach internetowych. Najbardziej znaną przeglądarką korzystającą z silnika jest Apple Safari, a w przeszłości używała go również Google Chrome. Jest to jeden z najpopularniejszych obecnie używanych silników renderujących. Celem tego projektu open source jest, aby WebKit stał się uniwersalnym silnikiem interakcji i prezentacji8 dla aplikacji. Oprócz wykorzystania w przeglądarkach internetowych, silnik wykorzystywany jest w różnego rodzaju oprogramowaniu, w tym w klientach pocztowych i komunikatorach internetowych.

Trident

Silnik renderujący Microsoftu nazywa się MSHTML lub, częściej, Trident. Nie jest niespodzianką, że Trident to silnik o zamkniętym kodzie źródłowym, który można znaleźć w Internet Explorerze. Jest to drugi najpopularniejszy silnik renderujący. Podobnie jak WebKit, Trident jest również używany w oprogramowaniu innym niż przeglądarki internetowe. Jednym z przykładów jest Google Talk.

Oprogramowanie może korzystać z silnika za pomocą biblioteki mshtml.dll, która jest dostarczana z systemem Windows. Trident pojawił się po raz pierwszy w wersji 4 przeglądarki internetowej i od tego czasu jest podstawą Internetu. Najnowsza wersja Internet Explorera firmy Microsoft nadal wykorzystuje Trident jako główny silnik renderujący.

Gecko

Firefox to najbardziej znany program, który wykorzystuje silnik renderujący Gecko o otwartym kodzie źródłowym. Jest to prawdopodobnie trzeci najpopularniejszy silnik renderujący za WebKit i Trident. Gecko to silnik renderujący typu open source, początkowo opracowany przez firmę Netscape w latach 90. dla przeglądarki internetowej Netscape Navigator. Nowoczesne wersje Gecko znajdują się głównie w aplikacjach opracowanych przez Mozilla Foundation i Mozilla Corporation, w szczególności w przeglądarce internetowej Firefox.

Presto

Presto jest (w momencie pisania tej książki) silnikiem renderującym dla Opery. Jednak w 2013 roku zespół Opery publicznie ogłosił, że wkrótce porzuci swój własny silnik renderujący Presto i przeprowadzi migrację do pakietu WebKit Chromium. Nazwa pakietu WebKit Chromium została następnie zmieniona na Blink (omówione w następnej sekcji). W żadnym innym momencie duża przeglądarka nie zmieniła kursu ze swoim silnikiem renderującym w tak dramatyczny sposób. To prawie na pewno oznacza wyginięcie Presto i stanie się jedną z ostatnich ofiar wojen przeglądarkowych.

Blink

W 2013 roku Google ogłosił, że rozwidla WebKit, aby stworzyć nowy silnik renderujący Blink. Początkowym celem Blink jest lepsze wsparcie wieloprocesowej architektury Chrome i zmniejszenie złożoności w przeglądarce. Czas pokaże, czy ten silnik będzie działał równie dobrze jak WebKit, ale sugestia, że Google usunie zbędną funkcjonalność, to dobry początek.

Geolokalizacja

Geolokalizacja API zapewnia urządzeniom mobilnym i komputerom stacjonarnym dostęp do lokalizacji geograficznej przeglądarki internetowej. Osiąga to za pomocą różnych metod, w tym GPS, triangulacji witryn komórkowych, geolokalizacji IP i lokalnych punktów dostępu Wi-Fi. Istnieje wiele oczywistych przypadków, w których te informacje mogą być nadużywane w rzeczywistych scenariuszach. Wprowadzono rygorystyczne zabezpieczenia przeglądarek w celu ograniczenia eksploatacji, pozostawiając główny wektor ataków jako socjotechnikę.

Pamięć internetowa

Magazyn sieciowy, czasami określany jako magazyn DOM, był częścią specyfikacji HTML5, ale już nim nie jest. Pomocne może być wyświetlenie pamięci internetowej jako doładowanych plików cookie. Podobnie jak pliki cookie, istnieją dwa główne typy przechowywania: jeden, który jest utrzymywany lokalnie i jeden, który jest dostępny podczas sesji. W przypadku przechowywania w Internecie pamięć lokalna utrzymuje się przez wiele wizyt użytkownika, a pamięć sesji jest dostępna tylko na karcie, która ją utworzyła. Jedną z głównych różnic między plikami cookie a przechowywaniem w sieci jest to, że przechowywanie w sieci jest tworzone wyłącznie przez JavaScript, a nie przez nagłówki HTTP, ani nie są one przesyłane do serwera w każdym żądaniu. Przechowywanie w sieci Web pozwala na znacznie większe rozmiary niż konwencjonalne pliki cookie. Rozmiar zależy od przeglądarki, ale zazwyczaj

wynosi co najmniej 5 megabajtów. Inną ważną różnicą jest to, że nie ma koncepcji ograniczeń ścieżek w przypadku przechowywania lokalnego.

PRZECHOWYWANIE SESJI

Oto prosty przykład korzystania z interfejsu API do przechowywania danych w sieci Web. Uruchom następujące polecenia w konsoli JavaScript przeglądarki internetowej. Ustawią wartość „BHH" w magazynie sesji bieżącej karty:

```
sessionStorage.setItem("BHH", "http://browserhacker.com");

sessionStorage.getItem("BHH");
```

SOP ma zastosowanie do przechowywania lokalnego, z podziałem na poszczególne miejsca pochodzenia. Inne źródła nie mogą uzyskać dostępu do magazynu lokalnego, ani subdomeny nie mogą uzyskać do niego dostępu.

Udostępnianie zasobów między źródłami

Współużytkowanie zasobów między źródłami (Cross-origin Resource sharing) lub CORS to specyfikacja, która zapewnia metodę dla źródła, która ignoruje SOP. W swojej najłagodniejszej konfiguracji aplikacja internetowa może zezwolić XMLHttpRequest z różnych źródeł na dostęp do wszystkich swoich zasobów z dowolnego źródła. Nagłówki HTTP informują przeglądarkę, czy jest dozwolony dostęp. Podstawowym składnikiem CORS jest dodanie do serwera WWW następujących nagłówków odpowiedzi HTTP:

```
Access-Control-Allow-Origin: *

Access-Control-Allow-Methods: POST, GET
```

Gdy przeglądarka wyśle cross-origin XMLHttpRequest do serwera, który nie odpowiada tymi nagłówkami, nie zostanie udzielony dostęp do treści odpowiedzi. Jest to zgodne z oczekiwanym zachowaniem SOP. Jeśli jednak serwer sieci Web zwróci poprzednie nagłówki, nowoczesne przeglądarki będą honorować specyfikację CORS i zezwolić na dostęp do zawartości odpowiedzi źródła.

HTML5

Jest bardzo prawdopodobne, że przeglądarka, której używasz, obsługuje teraz wiele elementów ze specyfikacji HTML5. HTML5 to kolejny standard HTML. Definiuje dodatki do specyfikacji, które zwiększają funkcjonalność, a co za tym idzie, wrażenia użytkownika w sieci. Oczywistą zmianą z punktu widzenia bezpieczeństwa jest zwiększenie powierzchni ataku. Zapewnia o wiele więcej metod, które nie miały ekspozycji na poprzednie Generacje HTML4. Zwiększa również permutacje, dzięki którym można korzystać z funkcji. Obie te kombinacje zwiększają ryzyko udanych ataków. Dotyczy to większości postępów w technologii i samo w sobie nie powinno być powodem, by nie robić postępów.

WebSocket

WebSocket to technologia przeglądarki, która umożliwia otwarcie interaktywnego i bardzo responsywnego, pełnodupleksowego kanału komunikacji między przeglądarką a serwerem. To zachowanie pozwala na stosowanie rygorystycznych akcji sterowanych zdarzeniami bez wyraźnej potrzeby odpytywania serwera. WebSocket jest zamiennikiem innych technologii AJAX-Push, takich jak Comet. Podczas gdy Comet wymaga dodatkowych bibliotek klienckich, interfejs API WebSocket jest implementowany natywnie w nowoczesnych przeglądarkach. Wszystkie najnowsze przeglądarki, w

tym Internet Explorer 10, obsługują standard WebSocket. Jedynymi wyjątkami są niektóre przeglądarki mobilne, takie jak Opera Mini i natywna przeglądarka Androida.

Web Workers

Przed robotami sieciowymi JavaScript w przeglądarce był środowiskiem jednowątkowym. Deweloperzy używaliby setTimeout() i setInterval(), aby osiągnąć wykonanie podobne do współbieżności. HTML5 wprowadza Web Workerss, które mogą być postrzegane jako wątki przeglądarki, ponieważ działają w tle. Istnieją dwa typy: jeden jest współdzielony przez wszystko, co działa w źródle, a drugi komunikuje się tylko z funkcją, która go utworzyła. Interfejs API ma różne inne ograniczenia, ale pracownicy sieci Web zapewniają programistom większą elastyczność. Ta sama elastyczność jest zapewniona atakującym, dając im więcej opcji wdrażania ataku w przeglądarce internetowej.

Manipulacja historią

Różne ataki opisane tu są wymierzone w funkcjonalność historii przeglądarki internetowej. Możliwości historii wciąż się zmieniają, ponieważ zmienia się zapotrzebowanie na przeglądarkę internetową. W przeszłości wystarczyło śledzić historię, gdy użytkownicy kliknęli w link, który przeniósł ich na inną stronę. Dzisiaj kliknięcie linku może wykorzystywać skrypty do renderowania strony, a to jest traktowane jako kamień milowy w doświadczeniu użytkownika. HTML5 oferuje metody manipulowania stosem historii. Skrypty mogą dodawać lub usuwać lokalizacje za pomocą obiektu historii, a także mogą przenosić bieżącą stronę do przodu lub do tyłu w łańcuchu historii.

WebRTC

Interfejs API komunikacji w czasie rzeczywistym (WebRTC) to znaczący rozwój, który wykorzystuje możliwości HTML5 i JavaScript. Umożliwia przeglądarkom komunikowanie się ze sobą z małym opóźnieniem i dużą przepustowością niezbędną do obsługi bogatej w multimedia komunikacji w czasie rzeczywistym. W chwili pisania tego tekstu WebRTC jest obsługiwany w najnowszych przeglądarkach Chrome, Firefox i Opera i jest do nich natywnie wbudowany. Udostępnia funkcje, takie jak bezpośredni dostęp do kamery i sprzętu audio (w celu obsługi wideokonferencji). Potencjalne implikacje bezpieczeństwa dla tego typu wysoce użytecznej, ale inwazyjnej technologii są oczywiste. Na szczęście WebRTC jest oprogramowaniem typu open source, więc nie jest poza zasięgiem przejrzystej analizy.

Podatności

Termin „podatności" jest abstrakcyjnym zbiorem, a zatem złożonym tematem. Można wywnioskować, że istnienie tego tekstu wynika wyłącznie z istnienia tak zwanych „podatności". Jednak definicja tego, co jest, a co nie jest podatnością, nie zawsze jest jasna. Czasami luka w zabezpieczeniach jest tak naprawdę częścią funkcjonalności, zgodnie z pierwotnym zamierzeniem, ale później okazuje się, że jest zbyt liberalna. Co gorsza, niektóre klasy podatności mają wiele nazw. W sumie cała sytuacja może być zagmatwana. Tutaj luki są wyjaśnione w kontekście ataku w celu jasności.

Presje ewolucyjne

Przeglądarki internetowe przeszły jedną z najbardziej dramatycznych i ekscytujących ewolucji w branży technologii informatycznych. Obecnie przeglądarki internetowe wykorzystują najnowocześniejsze techniki w zakresie wydajności, bezpieczeństwa i rozwoju. Przetrwają lub giną na niezwykle agresywnym polu bitwy. Przeglądarki internetowe były kiedyś znacznie mniej wyrafinowanym oprogramowaniem. Pierwsze manifestacje przeglądarki internetowej miały prosty cel - były wyświetlaniem i podążaniem za hiperłączami w embrionalnej sieci. Teraz mają wsparcie dla dodatków,

wtyczek, kamer, mikrofonów i geolokalizacji. Nie trzeba dodawać, że to długa droga od miejsca, w którym zaczęli. Krajobraz udziału w rynku przeglądarek internetowych był daleki od stałego w całej barwnej historii przeglądarki. Byli zwycięzcy i przegrani, przeglądarki niszowe i mainstreamowe, a reputacja rosła i spadała. Netscape był wczesną ofiarą bitew przeglądarek, ale jego upadek dał początek Organizacji Mozilla, a ostatecznie Firefox. Dawny Internet Explorer, niegdyś dominujący na rynku przeglądarek i zwycięzca Netscape′a, stale traci grunt na rzecz przeglądarek open source, a w ostatnich latach do komercyjnych ofert, takich jak Google Chrome i Apple Safari. Jednak ze względu na ciągły rozwój i podstawę finansowego giganta Microsoftu, nadal przetrwa i ewoluuje. Można śmiało powiedzieć, że ta opowieść o wojnie jeszcze się nie skończyła. Pole bitwy się zmieniło, a przeglądarki ewoluowały, aby stawić czoła nowemu terenowi. Ważnym wynikiem tego wyścigu zbrojeń jest to, że producenci przeglądarek rozumieją, że bezpieczeństwo jest ważne dla ich użytkowników i nieustannie utrudniają eksploatację przeglądarek. Doprowadziło to do różnych postępów w technologiach defensywnych. Poniższe sekcje opisują niektóre z głównych funkcji bezpieczeństwa przeglądarek obecnych w dzisiejszym ciężkim oprogramowaniu do obrony.

Nagłówki http

Duża część ewolucji bezpieczeństwa przeglądarki nastąpiła w nagłówkach HTTP. Ponieważ dyrektywy w zakresie całego żądania lub odpowiedzi są umieszczane w nagłówkach HTTP, stanowią one naturalny mechanizm dla serwera, który instruuje przeglądarkę, aby wprowadziła dodatkowe kontrole bezpieczeństwa.

Polityka bezpieczeństwa treści

CSP został zaprojektowany w celu złagodzenia luk XSS poprzez zdefiniowanie rozróżnienia między instrukcjami a treścią. Nagłówek HTTP CSP Content-Security-Policy lub X-Content-Security-Policy jest wysyłany z serwera w celu określenia lokalizacji, w których można załadować skrypty. Określa również ograniczenia dotyczące tych skryptów; na przykład, czy można użyć funkcji JavaScript eval().

Flaga bezpiecznych plików cookie

W przeszłości pliki cookie były wysyłane zarówno przez HTTP, jak i HTTPS bez rozróżniania między tymi dwoma źródłami. Może to mieć wpływ na bezpieczeństwo sesji nawiązanej z przeglądarką internetową. Token sesji bezpiecznie ustanowiony między serwerem a przeglądarką za pośrednictwem protokołu HTTPS może zostać ujawniony osobie atakującej za pomocą standardowego żądania HTTP. W tym miejscu flaga bezpiecznego pliku cookie przeskakuje za jednym zamachem wysokie budynki. Głównym celem tej flagi jest poinstruowanie przeglądarki, aby nigdy nie wysyłała pliku cookie przez żaden niezabezpieczony kanał. W ten sposób poufny token sesji może pozostać zamknięty w zaszyfrowanej barierze za każdym razem, gdy jest w trakcie przesyłania.

Flaga HttpOnly plików cookie

Flaga HttpOnly to kolejna opcja, którą można zastosować do plików cookie, a wszystkie nowoczesne przeglądarki przestrzegają tej dyrektywy. Flaga HttpOnly instruuje przeglądarkę, aby zabroniła dostępu do zawartości pliku cookie z dowolnych skryptów. Ma to zaletę bezpieczeństwa polegającą na ograniczaniu kradzieży plików cookie wynikających z XSS z JavaScript.

X-Content-Type-Options

Przeglądarki mogą wykorzystywać różne metody wykrywania treści, aby zgadnąć, jaki typ treści został zwrócony z serwera WWW. Na tej podstawie przeglądarka wykona odpowiednią akcję, która jest zmapowana do tego typu zawartości. Dyrektywa nosniff istnieje, aby wyłączyć tę funkcjonalność i

zmusić przeglądarkę do renderowania treści zgodnie z nagłówkiem content-type. Na przykład, jeśli serwer wyśle dyrektywę nosniff w odpowiedzi na tag skryptu, przeglądarka zignoruje odpowiedź, chyba że typ MIME pasuje aplikacja/javascript (i kilka innych). W witrynie takiej jak Wikipedia (która zezwala na przesyłanie) może to być szczególnie niepokojące. Brak dyrektywy staje się problemem, gdy specjalnie spreparowany plik jest przesyłany, a następnie pobierany. Przeglądarka może zostać nakłoniona do nieprawidłowego zinterpretowania typu MIME danych i zinterpretowania na przykład pliku JPEG jako skryptu. Ma to oczywiste problemy przy rozważaniu kontroli bezpieczeństwa przeglądarki; może być możliwe, aby użytkownik przejął kontrolę nad przeglądarką za pośrednictwem publicznej aplikacji internetowej. Jednym ze sposobów byłoby przesyłanie plików o dozwolonym (i pozornie bezpiecznym) typie treści, które są następnie interpretowane w inny, bardziej niebezpieczny i niestabilny sposób.

Ścisłe bezpieczeństwo transportu

Ten nagłówek HTTP informuje przeglądarkę, że komunikacja z witryną musi odbywać się przez prawidłowy tunel HTTPS. Użytkownik nie będzie mógł zaakceptować błędów HTTPS i przejść przez niezabezpieczone połączenie. Zamiast tego przeglądarka wyjaśni błąd, nie pozwalając użytkownikowi na kontynuowanie przeglądania.

X-Frame-Option

Nagłówek HTTP X-Frame-Options służy do zapobiegania ramkom strony w przeglądarce internetowej. Gdy przeglądarka widzi nagłówek, powinna upewnić się, że wysłana strona nie zostanie wyświetlona w ramce IFrame. Ten nagłówek został opracowany w celu zapobiegania atakom polegającym na przywracaniu interfejsu użytkownika, z których jednym jest Clickjacking. Atak ten polega na umieszczeniu strony ofiary w ramce w oknie na pierwszym planie, które jest w 100% przezroczyste. Użytkownicy uważają, że wchodzą w interakcję z nieprzezroczystą stroną tła (atakującego), ale w rzeczywistości klikają niewidoczną stronę pierwszego planu (ofiarę). Nagłówek HTTP X-Frame-Options uniemożliwia pomyślne wykonanie podzbioru ataków związanych z przeadresowaniem interfejsu użytkownika.

Odbite filtrowanie XSS

Jest to funkcja zabezpieczeń przeglądarki internetowej, która próbuje wykryć, oczyścić i zablokować odbity XSS . Przeglądarka internetowa próbuje pasywnie wykryć udaną eksploatację Reflected XSS. Następnie próbuje oczyścić skrypty dostarczone w odpowiedzi i, w większości przypadków, uniemożliwia ich wykonanie.

Sandboxing

Sandboxing to próba rozwiązania rzeczywistego problemu w świecie rzeczywistym. Podstawowym założeniem jest to, że przeglądarka zostanie naruszona i znajdzie się pod kontrolą atakującego. Nigdy nie padły prawdziwsze słowa! Fundamentalne (i pragmatyczne) stanowisko jest takie, że programiści nieuchronnie napiszą podatny na ataki kod. Wiele osób uważa, że zagrożony kod nieuchronnie pojawi się gdzieś w oprogramowaniu. Spójrzmy prawdzie w oczy, nawet ci ze społeczności bezpieczeństwa, którzy wskazują palce na programistów, są podatni. Piaskownica to dobra próba rozwiązania tego uniwersalnego problemu. Oczywiście stopień, w jakim programiści zastosują się do tego założenia (tj. napiszą podatny na ataki kod) będzie się różnić w zależności od wielu złożonych czynników, takich jak brak snu czy jakość ziaren kawy. Piaskownica to po prostu kontrola łagodząca. Próbuje zamknąć obszar wysokiego prawdopodobieństwa naruszenia bezpieczeństwa przeglądarki w ścianie ochronnej. Pozwala na większe skupienie się na mniejszej powierzchni ataku. Zapewnia to dobrą inwestycję

zasobów typu „ryzyko w stosunku do nagrody” dla zespołu ds. bezpieczeństwa przeglądarek. Sandboxing nie jest nowym rozwiązaniem; różnice zaobserwowano w innych obszarach informatyki. Na przykład Sun używał podziału na segmenty w Trusted Solaris, a FreeBSD używał Jails. To ograniczało dostęp do zasobów w zależności od uprawnień procesu.

Piaskownica przeglądarki

Piaskownica może być stosowana na wielu poziomach. Może na przykład zostać zastosowany na poziomie jądra, aby oddzielić jednego użytkownika od innego użytkownika. Można go zastosować na poziomie sprzętu, aby osiągnąć separację uprawnień między jądrem a przestrzenią użytkownika. Piaskownica przeglądarki jest piaskownicą najwyższego poziomu możliwą dla programu w przestrzeni użytkownika. Jest to bariera między uprawnieniami nadanymi przeglądarce przez system operacyjny a uprawnieniami podprocesu działającego w przeglądarce. Aby całkowicie zhakować przeglądarkę, musisz wykonać co najmniej dwa kroki. Pierwszym z nich jest znalezienie luki w funkcjonalności przeglądarki. Następnym krokiem jest przebicie się przez piaskownicę. Ten ostatni jest znany jako obejście piaskownicy. Niektóre strategie piaskownicy przeglądarek otwierają każdą witrynę internetową w oddzielnych procesach, co utrudnia złośliwej witrynie wywieranie dalszego wpływu na inne aktualnie odwiedzane witryny lub na sam system operacyjny. Ta piaskownica dotyczy również wtyczek i rozszerzeń, takich jak oddzielne przetwarzanie dla renderowania PDF. Luki w zabezpieczeniach związane z obejściem piaskownicy są zwykle skompilowanym kodem i próbują całkowicie podważyć funkcjonalność uruchomionego procesu. Na tym etapie testowana jest skuteczność piaskownicy: czy może ona uniemożliwić podwróconej ścieżce wykonania uzyskanie pełnych uprawnień do procesu?

Piaskownica IFrame

Ramki IFrame mogą być używane jako mechanizm dołączania potencjalnie niezaufanych treści z zasobów o różnych źródłach, a w niektórych przypadkach niezaufanych treści z zasobów tego samego pochodzenia. Na przykład jednym z popularnych elementów w witrynach internetowych jest widżet mediów społecznościowych Facebooka. Możliwość, że ramka IFrame stanie się nieprzyjazna, nie jest nowym pomysłem, a dostawcy przeglądarek od dawna oferują różne sposoby na złagodzenie szkód ubocznych spowodowanych przez nieuczciwą ramkę IFrame. Specyfikacja HTML5 przedstawiła propozycję sandboxingu IFrame, która została przyjęta przez nowoczesne przeglądarki. Daje to programistom sposób aby zatrudnić jak najmniej przywilejów. Ramki IFrames w trybie piaskownicy to metoda dołączania atrybutu HTML5, który dodaje dodatkowe ograniczenia do ramki wbudowanej. Ograniczenia te obejmują uniemożliwienie korzystania z formularzy, zatrzymanie wykonywania skryptu, uniemożliwienie nawigacji u góry i uwięzienie go w miejscu pochodzenia. Ograniczenia te rozciągają się na każdą ramkę nadrzędną, zapewniając, że wszelkie zagnieżdżone ramki IFrame automatycznie odziedziczą ograniczenia po utworzeniu.

Ochrona przed phishingiem i złośliwym oprogramowaniem

Fałszowanie podmiotów online (w tym wiadomości e-mail) w celu kradzieży danych osobowych, takich jak dane uwierzytelniające, jest tradycyjnie nazywane phishingiem. Wiele organizacji posiada usługi katalogowania znanych witryn phishingowych, a nowoczesne przeglądarki mogą wykorzystywać te informacje. Przeglądarka sprawdza każdą odwiedzaną witrynę pod kątem znanej listy złośliwych witryn. Jeśli wykryje, że żądana witryna jest w rzeczywistości witryną phishingową, przeglądarka podejmie działanie. Podobnie serwery internetowe mogą zostać zainfekowane bez zgody właściciela lub są tworzone specjalnie w celu przechowywania treści, które mogą próbować złamać przeglądarkę, wykorzystując znane luki w zabezpieczeniach. Witryny te mogą również zachęcać użytkownika do ręcznego pobierania i uruchamiania oprogramowania, które ominie zabezpieczenia przeglądarki i

zostanie uruchomione bezpośrednio. Różne organizacje utrzymują aktywne czarne listy witryn, w których udowodniono, że zawierają złośliwy kod, i mogą być połączone bezpośrednio z przeglądarką w celu zapewnienia ochrony w czasie rzeczywistym

Zawartość mieszana

Witryny internetowe z lukami dotyczącymi zawartości mieszanej pochodzą ze schematu HTTPS, a następnie żądają zawartości za pośrednictwem protokołu HTTP. Oznacza to, że wszystko, co składa się na tworzenie strony, nie jest dostarczane przez HTTPS. Dane, które nie są przesyłane przez HTTPS, są zagrożone modyfikacją i mogą zniweczyć jakąkolwiek zaletę szyfrowania niektórych danych. W przypadku przesyłania skryptu przez niezaszyfrowany kanał osoba atakująca może wstrzyknąć do strumienia danych instrukcje, które zagrażają interakcji między przeglądarką internetową a aplikacją internetową.

Podstawowe problemy z bezpieczeństwem

Ewolucja stale rozszerzającego się zestawu funkcji kontroli bezpieczeństwa przeglądarki stanowi podstawę szerszego i bardziej fundamentalnego obrazu. Tradycyjne zabezpieczenia sieci opierały się na wdrażaniu i utrzymywaniu zabezpieczeń zewnętrznych lub obwodowych, takich jak zapory sieciowe. Z biegiem czasu zaobserwowano, że urządzenia te blokują cały ruch poza zasadniczym nie tylko do organizacji, ale także z niej. Chociaż sieć staje się coraz ciaśniejsza, firmy nadal potrzebują dostępu do swoich informacji, a wzrost wykorzystania technologii internetowych (prawie wszystkiego, co podróżuje przez port TCP 80 lub 443) rośnie w coraz szybszym tempie. W rzeczywistości zapory ogniowe tak skutecznie ograniczają otwarte zalewy ruchu, że często pozostaje nam jedynie promień ruchu HTTP. Dobrym tego przykładem jest wzrost popularności technologii SSL VPN w stosunku do tradycyjnych sieci IPSEC VPN. Prawdopodobnie wszystkie zapory ogniowe skutecznie ograniczyły ruch sieciowy do dwóch portów: 80 i 443. Przenosi to skrajne zaufanie na model bezpieczeństwa przeglądarki internetowej. Poniższe podrozdziały przedstawiają ogólny obraz dotyczący bezpieczeństwa przeglądarek i wyjaśniają, dlaczego działające przeciwstawne siły tworzą złożony plac zabaw ataku i obrony. Zbiegamy się, dlaczego, ogólnie rzecz biorąc, wiązka laserowa ruchu internetowego nie odizolowała obwodu sieci, a zamiast tego stworzyła pryzmat możliwości ataku.

Powierzchnia ataku

Znaczenie powierzchni ataku prawdopodobnie nie będzie dla ciebie nowe. Powierzchnia ataku to region przeglądarki, który jest podatny na wpływy z niezaufanych źródeł. Biorąc pod uwagę, że w najmniejszym przypadku jest to cały silnik renderujący, skala problemu staje się jasna. Przeglądarka internetowa ma dużą i stale rosnącą powierzchnię ataku. Istnieje szeroka gama interfejsów API i liczne abstrakcje do przechowywania i przywoływania danych. Odwrotnie, powierzchnia ataków w całej sieci może być teraz pod ścisłą kontrolą. Punkty dostępu i dozwolone przepływy ruchu są dobrze zrozumiane, a procesy kontroli zmian mogą uwzględniać zmiany. Na przykład dostęp do różnych portów w zaporze można w prosty sposób zweryfikować i ograniczyć za pomocą dobrze znanych metod. Rzadko zdarza się, aby dostawca przeglądarki usuwał funkcje z oprogramowania. Coraz częściej sprzedawcy dodają najnowsze gadżety. Podobnie jak w przypadku większości produktów, rzadko występuje widoczna nagroda za zmniejszenie zdolności, podczas gdy zachowana jest kompatybilność wsteczna. Ponieważ zestaw funkcji jest rozszerzony, zwiększa się również wielkość potencjalnej powierzchni ataku. Nowoczesne przeglądarki aktualizują się automatycznie i po cichu w tle, czasami zmieniając powierzchnię ataku bez wiedzy obrońcy. W niektórych przypadkach może to być dobre. Jednak dla dojrzałego i zdolnego zespołu ds. bezpieczeństwa może to stanowić więcej wyzwań niż korzyści. Jeśli jednak chodzi o zwykłą przeglądarkę internetową, rzadko można znaleźć członków zespołu ds. bezpieczeństwa organizacji z dużym doświadczeniem w jej obronie. Mimo że to pojedyncze

oprogramowanie jest jednym z najbardziej zaufanych, potencjalnie stanowi największą powierzchnię ataku w Internecie.

Tempo zmian

Zespoły ds. bezpieczeństwa przeglądarek mogą nie pracować na osi czasu, która jest zgodna z organizacją. Często organizacja nie ma kontroli nad wdrażaniem poprawek przeglądarki, które mogą być pożądane w celu wzmocnienia stanu bezpieczeństwa. Błędy przeglądarki internetowej związane z bezpieczeństwem są często traktowane przez programistów z niższym priorytetem niż niektórzy członkowie społeczności zajmującej się bezpieczeństwem. W wydaniu Firefoksa 18.0 ze stycznia 2013 r. Mozilla pochwaliła się, że jedną z poprawek było zapobieganie lukom w zabezpieczeniach zawartości mieszanej. Oznacza to wyłączenie ładowania treści HTTP, gdy źródło ma schemat HTTPS. Możesz być zaskoczony, gdy dowiesz się, że ten błąd został po raz pierwszy zgłoszony w grudniu 2000 r. Jest to prawdopodobnie najgorszy przypadek, ale służy do zademonstrowania opóźnienia, które może wystąpić. Brak kontroli użytkownika końcowego nad aktualizacjami zabezpieczeń przeglądarki internetowej nie różni się od innych elementów oprogramowania. Jest również mało prawdopodobne, aby organizacja była w stanie zatrzymać korzystanie z każdej przeglądarki w oczekiwaniu na krytyczną poprawkę. Jeśli to założenie się utrzyma, większość organizacji będzie podatna na ataki na przeglądarkę w przedziale czasowym między wydaniem publicznego exploita a dostawcą poprawki.

Cicha aktualizacja

Ciche aktualizacje w tle, oferujące potencjalną drogę ataku, zapewniają również prawdopodobnie większą wartość dla użytkowników. Konieczność zapewnienia szybkiego stosowania dostępnych aktualizacji skłoniła niektórych programistów do wdrożenia własnych cichych mechanizmów. Google na przykład wdrożył funkcję cichej aktualizacji w swojej przeglądarce Chrome. Użytkownik nie miał możliwości wyłączenia tej funkcji, dzięki czemu wszystkie aktualizacje zostały zastosowane w odpowiednim czasie bez interwencji użytkownika. Jednym z godnych uwagi przykładów było wykorzystanie cichej aktualizacji przez Google do wdrożenia własnego silnika renderowania plików PDF w przeglądarce Chrome w celu zastąpienia oprogramowania Adobe Reader. Dzięki temu każda samoaktualizująca się instancja Chrome nie była już objęta procesem aktualizacji tej wtyczki innej firmy. Teraz w tym tkwi problem. Przeglądarki aktualizujące i dodające funkcje w tle potencjalnie zwiększają powierzchnię ataku każdej przeglądarki, jeśli zostaną wykonane nieprawidłowo. Wymaga to również, aby zespół ds. bezpieczeństwa dowolnej organizacji zlecił pewien stopień zależności programistom przeglądarki. W połączeniu z faktem, że obszary, na które zwraca się uwagę programistów przeglądarki, mogą nie być zgodne z potrzebą danej organizacji użytkownika końcowego, ta zależność może być frustrująca.

Rozszerzenia

Rozszerzenia zapewniają metodę rozszerzania zachowania przeglądarki bez korzystania z samodzielnego oprogramowania. Mogą wpływać na każdą stronę, którą ładuje przeglądarka i odwrotnie - każda strona może na nie potencjalnie wpływać. Każde rozszerzenie dodaje miejsce, w które haker może celować, a tym samym zwiększa powierzchnię ataku przeglądarki. W niektórych przypadkach dla tej przeglądarki można nawet wprowadzić uniwersalne luki XSS. Zagłębiasz się w rozszerzenia i ich luki dalej w Części 7.

Wtyczki

Wtyczka to zazwyczaj oprogramowanie, które może działać niezależnie od przeglądarki. W przeciwieństwie do rozszerzeń przeglądarka uruchamia wtyczki tylko wtedy, gdy aplikacja internetowa

zawiera je na stronie za pomocą znacznika obiektu lub, w niektórych przypadkach, nagłówka typu content. Niektóre strony internetowe nie mogą być dostępne bez odpowiednich wtyczek i dlatego przeglądarki zapewniają możliwość zwiększania ich funkcjonalności. Na przykład aplety Java są używane w niektórych bramach VPN, takich jak Juniper. Wiele popularnych wtyczek do przeglądarek jest wymaganych w przypadku standardowej działalnościi, a niektóre z tych wtyczek zawierały w przeszłości luki w zabezpieczeniach. Oznacza to, że obrońca musi podjąć decyzję o użyciu podatnego oprogramowania lub wyłączeniu części działalności biznesowej. Większość wtyczek nie ma centralnego mechanizmu aktualizacji. Oznacza to, że w niektórych przypadkach zabezpieczenia należy wprowadzać ręcznie. Oczywiście powoduje to obciążenie i złożoność obrony infrastruktury. Wtyczki mają zwykle negatywny wpływ na media bezpieczeństwa. Wiele z tych aplikacji ma poważne luki w zabezpieczeniach, a w niektórych przypadkach okazuje się, że ich bezpieczeństwo jest tak duże, że specjaliści ds. bezpieczeństwa doradzają organizacjom całkowite ich usunięcie. Producenci systemów operacyjnych również działali niezależnie, dezaktywując wrażliwe wtyczki za pomocą własnych automatycznych schematów aktualizacji, przez czas nieokreślony lub do momentu znalezienia rozwiązania. Wtyczki mogą zwiększyć powierzchnię ataku. Ujawniają dodatkowe funkcje i cele dla hakera. Więcej szczegółów na temat wtyczek znajdziesz w Części 8.

Poddanie kontroli

Przeglądarka żąda instrukcji z dowolnych lokalizacji w Internecie. Jego podstawową funkcją jest renderowanie treści na ekran i udostępnianie interfejsu użytkownika dla tej treści, dokładnie tak, jak zamierzał autor. Jako produkt uboczny tej podstawowej funkcji, konieczne jest przekazanie znacznego stopnia kontroli serwerowi WWW. Przeglądarka musi wykonać dostarczone polecenia, w przeciwnym razie może nie zostać poprawnie wyrenderowana strona. We współczesnej sieci często aplikacja internetowa zawiera wiele zasobów i skryptów z innych źródeł. Te również muszą zostać wykonane, jeśli strona ma być wyświetlana zgodnie z przeznaczeniem. Tradycyjnie te instrukcje mogły być tak proste, jak: „Gdzie powinienem umieścić ten tekst i dokąd ten obraz?" Z drugiej strony nowoczesne aplikacje internetowe i przeglądarki mogą żądać: „Włączę teraz twój mikrofon i wyślę te dane asynchronicznie na serwer". Ten rodzaj inwazyjnej funkcjonalności natychmiast rodzi pytanie, czy wszyscy użytkownicy mają gwarancję, że będą przeglądać tylko niezłośliwe strony internetowe. Odpowiedź brzmi prawie we wszystkich okolicznościach, oczywiście, że nie! Niemożność zagwarantowania w czasie rzeczywistym nienaruszalności treści pozyskiwanych z odległych lokalizacji jest podstawową podstawą wszystkich luk w zabezpieczeniach przeglądarek i ich wykorzystywania.

Kontrola protokołu TCP

Model serwer-klient nie jest często tak elastyczny, jeśli chodzi o port, na którym komunikuje się klient, lub adresy IP, z których klient może korzystać podczas wymiany danych. Ta funkcja może być bardzo przydatna dla atakującego. Oznacza to, że prawie nie ma ograniczeń dotyczących atakowania tylko protokołów HTTP lub określonych systemów. W grę wchodzą tutaj inne czynniki, które przygotowują grunt pod zupełnie nową klasę ataków. Zbadasz te ataki międzyprotokołowe w Części 10.

Szyfrowana komunikacja

SSL i TLS mogą być używane do komunikacji z zaufanymi organizacjami przez Internet, chroniąc integralność i poufność wiadomości za pomocą szyfrowania. I odwrotnie, dokładnie ta sama technologia może być również używana do bezpiecznej komunikacji z atakującymi. Celem szyfrowanej komunikacji między przeglądarką a serwerem jest ochrona danych między tymi dwoma punktami końcowymi. Stwarza to poważne komplikacje dla obrońców. Nie mają możliwości wykrycia szkodliwych danych. Ten szyfrowany tunel obsługiwany przez przeglądarkę działa na korzyść atakujących jak przemycają swoje polecenia i przemycają swoje łupy.

Zasady tego samego pochodzenia

SOP jest stosowany niespójnie w różnych technologiach przeglądarek i jest prawdopodobnie jedną z najbardziej mylących koncepcji. Jak wcześniej wspomniano, SOP został stworzony w celu odizolowania zasobów widocznych w przeglądarce, aby zapobiec interakcji elementów z jednej lokalizacji z innymi, niepowiązanymi zasobami pochodzącymi z innych lokalizacji, które również działają w tej samej przeglądarce. Zasadniczo jest to piaskownica. Ta konkretna piaskownica ma ogromne znaczenie dla bezpieczeństwa przeglądarki. Biorąc pod uwagę pierwszorzędną pozycję przeglądarki w centrum aktywności sieci, przeglądarka skutecznie łączy odmienne strefy zaufania w standardzie – i jest odpowiedzialna za utrzymanie spokoju. Aby wesprzeć potrzeby każdej strefy, autonomiczne funkcje, które mogą wchodzić w interakcje z pochodzeniem, są dość rozległe. Jeśli te funkcje mogą naruszyć SOP, legalne funkcje stają się wrogie, ponieważ mogą teraz przechodzić przez strefy bezpieczeństwa. Zrozumienie SOP nie kończy się na zrozumieniu jego implementacji w samej przeglądarce. Implementacje SOP często znacznie różnią się między przeglądarkami, ich wersjami, a nawet wtyczkami. Rozdział 4 zagłębia się bardzo głęboko w SOP we wszystkich jego wcieleniach i oferuje mnóstwo sposobów na ominięcie tej kontroli. Te obejścia są dostępne dzięki dziwakom SOP w Javie, Adobe Reader, Silverlight i różnych implementacjach przeglądarek.

Błędy

Wiele praktycznych zasad, które działały w przeszłości, nie ma już zastosowania w obecnym globalnym krajobrazie zagrożeń. Poniższe błędy są łatwymi pułapkami, w które można wpaść. Niestety, wiele z tych błędnych poglądów nadal jest propagowanych przez ludzi, którzy mają dobre intencje. Błąd zasady solidności Zasada solidności, znana również jako prawo Postela, nakazuje programistom „być konserwatywnym w tym, co robisz, być liberalnym w tym, co akceptujesz od innych”. Nie idzie to w parze z praktycznym bezpieczeństwem. Przeglądarka internetowa jest niezwykle liberalna w kwestii tego, co będzie renderować. Jest to jeden z głównych powodów, dla których XSS był tak trudny do wyrugowania. Przeglądarka utrudnia tworzenie bezpiecznych filtrów i koderów, ponieważ przeglądarka internetowa umożliwia wykonywanie instrukcji na wiele sposobów. Aby zachęcić programistów do bezpiecznych praktyk kodowania, Zasadę Solidności należy zastąpić sformułowaniem „Bądź konserwatywny w tym, co robisz, bądź ultra konserwatywny w tym, co akceptujesz od innych”. Gdyby zostało to zaszczepione następnej generacji programistów, hakerzy mieliby znacznie trudniejszy czas!

Błąd zewnętrznej granicy bezpieczeństwa

Wiele organizacji lubi abstrahować swoje granice bezpieczeństwa, aby wymyślić dostosowany model zamku i fosy. Będą rozumowali, że ich obrona ma pierścienie ścian, które chronią ich krytyczne zasoby. Błędne założenie jest takie, że warstwowe podejście do cebuli zapewnia najbezpieczniejsze rezultaty. Niestety nie jest to średniowieczna Europa. To złożona sieć! Podstawowym problemem związanym z tym schematem obrony jest to, że zakłada on, że atakujący wchodzą z najbardziej zewnętrznej warstwy i, w sposób Braveheart, walczą sekwencyjnie przez każdą ścianę. Pojęcie to odbiega od rzeczywistości prawie tak bardzo, jak filmy hollywoodzkie od prawdziwych wydarzeń historycznych. Intranet organizacji to stale ewoluujące środowisko, w którym w całej infrastrukturze pojawiają się napastnicy w stylu Whac-A-Mole. Rzeczywistość jest taka, że przeglądarka internetowa jest płodna i w niektórych przypadkach działa jak portal bezpośrednio przez zewnętrzny obwód. W związku z tym granice obronne zostały pośrednio naruszone i nie mogą bronić się przed atakami odbijającymi się od przeglądarki internetowej. Należy zainwestować środki obronne w Micro Security Perimeter, która musi obejmować aktywa krytyczne. Dzisiejsze sieci muszą bronić się przed zmianą urządzeń ze

sprzymierzeńca na wroga, kiedy najmniej się tego spodziewa. W prawdziwym świecie bezpieczeństwo to ograniczony zasób, który należy przeznaczyć tam, gdzie wzmocni obronę najcenniejszych zasobów.

Metodologia hakowania przeglądarki

W tym miejscu mamy nadzieję, że doceniasz złożoność wyzwań stojących przed przeglądarką. Zabezpieczanie sieci nie jest łatwym zadaniem i prawdopodobnie duża część odpowiedzialności za to spada na przeglądarkę. To pierwsza i ostatnia linia obrony. W hipotetycznym, postapokaliptycznym świecie zaawansowanych technologii, w którym każda witryna internetowa była zagrożona i złośliwa, idealna przeglądarka nadal zapewniałaby bezpieczeństwo komputera. Jesteśmy bardzo daleko od tej utopii bezpieczeństwa. Nadszedł czas, aby zdekonstruować niejasne pojęcie hakowania przeglądarki i przekształcić je w etapowe podejście, które może przetrwać poza eliminacją obecnych słabości i przetrwać redakcję. Zdefiniowaliśmy metodę, która, mamy nadzieję, utrzyma aktualność niezależnie od obecnego terenu bezpieczeństwa. Ta sekcja przedstawia naszą metodologię i proponowaną chronologię hakowania przeglądarki internetowej. Ta metodologia ma na celu skuteczne kierowanie działaniami hakerskimi przeglądarki. Części zostały uporządkowane tak, aby odwzorować bezpośrednio główne etapy metodologii. Każda Część skupia się na praktycznych etapach i zagłębia się w specyfikę techniczną. Gdy opanujesz każdą Część ł, zwiększysz swoje zrozumienie metodologii.

W zależności od celu, niektóre ścieżki w metodologii mogą być trywialne, ponieważ ogólnodostępne narzędzia bezpieczeństwa zautomatyzują proces. Inne części będą stanowić większe wyzwanie. Metodologia hakowania przeglądarki składa się z trzech głównych sekcji, które obejmują etapy hakowania wysokiego poziomu. Są one reprezentowane jako kropkowane linie otaczające różne fazy na diagramie. To pogrupowanie etapów zapewnia przegląd postępu metodologii, począwszy od inicjowania, przechodząc do zatrzymania, a następnie do ataku. Pierwsza enkapsulacja to Inicjowanie, czyli konfiguracja całego procesu. Następnie następuje enkapsulacja z zachowaniem i jest to miejsce, w którym odbywa się utrzymanie twojego zrozumienia przeglądarki. To jest stworzenie przyczółka w przeglądarce docelowej lub urządzeniu, na którym znajduje się przeglądarka; jest to inicjalizacja włamania do przeglądarki. Prawdziwa akcja pojawia się w następnym zgrupowaniu. Enkapsulacja Ataku zawiera siedem opcji ataku, które są omówione w kolejnych sekcjach i bardziej szczegółowo w dalszej części. Podczas tych faz różne aspekty przeglądarki będą atakowane i wykorzystywane. Niektóre z omówionych technik ataku mogą skutkować dodatkowymi fazami inicjowania w innych instancjach przeglądarki, co skutkuje cykliczną ekspansją ataku i zakresem włamań. Inicjowanie Enkapsulacja inicjująca ma w sobie jedną fazę. Ta pozornie nieszkodliwa faza jest pierwszym i najważniejszym krokiem w hackowaniu przeglądarki internetowej. Bez tej fazy żadne inne ataki nie są możliwe, a docelowa przeglądarka jest poza zasięgiem.

Inicjowanie kontroli

Każda permutacja sekwencji ataku rozpoczyna się od uruchomienia instrukcji w przeglądarce internetowej. Aby tak się stało, przeglądarka musi napotkać (i wykonać) instrukcje pod Twoją kontrolą. To jest temat w Części 2, który omawia metody, za pomocą których można oszukać, zwabić, oszukać lub zmusić przeglądarkę do napotkania i, co najważniejsze, wykonania dowolnego kodu.

Przytrzymanie

Teraz, po pomyślnym zaatakowaniu, jak zwiększyć kontrolę nad celem? Musisz zachować kontrolę nad przeglądarką w sposób ułatwiający dalsze przeprowadzanie ataków.

Zachowanie kontroli

Rozważmy fantazję o dżinach i trzech życzeniach; pojawia się dżin i spełnia trzy życzenia. Sprytny odbiorca może próbować poszerzyć swoje szczęście, życząc sobie więcej życzeń z ostatnim życzeniem - w ten sposób poddaj testowi politykę wykluczania dżina! Cóż, w przypadku utrzymywania komunikacji z zaatakowaną przeglądarką internetową, początkowy kod instruuje przeglądarkę, aby wielokrotnie prosiła o kolejne życzenie. Uwalniasz dżina na etapie zaczepiania i od tego momentu zniewalasz przeglądarkę, aby dalej spełniała życzenia. Tak jak dżin może zniknąć w obłoku dymu, tak ten stan rzeczy może nie trwać długo. Pozycja niekończących się życzeń zależy od działań, które następnie podejmuje użytkownik. Może zamknąć kartę, w której miało miejsce początkowe wykorzystanie, lub użyć jej do przejścia do innej witryny, zamykając w ten sposób ładunek JavaScript, a tym samym kanał komunikacji.

Zanim dasz się ponieść kolejnym atakom, dobrze jest uzbroić się w cierpliwość i zamiast tego zastanowić się nad metodami zwiększania wpływu na przeglądarkę. W tej fazie metodologii próbujesz zmniejszyć możliwość utraty kontroli nad przeglądarką przez użytkownika surfującego poza źródłem lub nawet zamykającego przeglądarkę internetową. Możesz to osiągnąć na kilka sposobów na różnych poziomach trwałości. Ważne jest, aby uzbroić się w cierpliwość i maksymalnie wykorzystać tę fazę przed przejściem do następnej, ponieważ im dłużej możesz trzymać przeglądarkę, tym większą powierzchnię ataku możesz przesłuchać i tym bardziej kontrolowany będzie twój atak. Warto również zauważyć, że czasami, podczas późniejszej enkapsulacji Ataku, udane ataki ujawnią metody zwiększenia siły przyczółka, poprawiając stopień kontroli. Z tego powodu istnieje dwukierunkowa strzałka między dwiema fazami w metodologii. Doświadczenie pomoże określić, gdzie wysiłki mające na celu zwiększenie odporności kanału kontroli powinny zastąpić wysiłki w zakresie atakowania, a gdzie wysiłki w zakresie atakowania zostały wykorzystane w elastyczności i trwałości kanału kontroli.

Napadający

Na tym etapie metodologii wykorzystujesz kontrolę uzyskaną nad przeglądarką i badasz możliwości ataku z obecnej pozycji. Ataki mogą przybierać różne formy, od „lokalnych” ataków na instancję przeglądarki lub system operacyjny, w którym się ona znajduje, po ataki na zdalne, odmienne systemy w dowolnych lokalizacjach. Uważny czytelnik zauważy, że Omijanie polityki tego samego pochodzenia znajduje się na szczycie i poza innymi elementami enkapsulacji atakującej. Dlaczego to jest takie? Ponieważ pasuje do wszystkich kroków ataku. Jest to kontrola bezpieczeństwa, która zostanie ominięta lub wykorzystana w innych fazach eksploatacji. Coś innego, co powinno szybko stać się oczywiste, to cykliczna strzałka w centrum enkapsulacji atakującego. Daleki od bycia po prostu rewolucyjnym, prawdopodobne jest, że którakolwiek z faz ataku ujawni szczegóły, które mogą doprowadzić do udanego ataku w którejkolwiek z pozostałych faz. Z tej pozycji prawdopodobnie będziesz przeskakiwać między różnymi kategoriami, w zależności od tego, która z nich najprawdopodobniej przyniesie najskuteczniejsze nagrody. Zdefiniowano siedem podstawowych klas ataków, które można uruchomić z przeglądarki internetowej. Przy podejmowaniu decyzji o tym, jaką drogę należy tutaj obrać, w grę wchodzą różne czynniki. Głównymi wpływami będą zakres zaangażowania, pożądany cel i możliwości podpiętej przeglądarki.

Omijanie zasad tego samego pochodzenia

SOP można traktować jako podstawową piaskownicę. Jeśli jesteś w stanie go ominąć, automatycznie utworzyłeś udany atak, mając dostęp do innego źródła, które zostało wcześniej zablokowane przez przeglądarkę. Omijając SOP, możesz teraz zaatakować nowo odkryte źródło za pomocą dowolnej innej możliwej do zastosowania techniki w potencjalnej reakcji łańcuchowej. Różnorodne interpretacje SOP zostały szczegółowo omówione w Części 4. W przypadku obejścia istnieje wiele ataków, które można

przeprowadzić bez zakłóceń. W tym rozdziale zbadasz niektóre niespójności i sposoby wykorzystania tego błędu w najbardziej podstawowej kontroli bezpieczeństwa przeglądarki.

Atakowanie użytkowników

Pierwsza opcja ataku przedstawiona w metodologii hakera przeglądarki to Atakowanie użytkowników, omówione w Części 5. Obejmuje ona ataki z udziałem użytkowników przeglądarki i ich potencjalnie niejawnego zaufania do środowiska kontrolowanego przez atakującego. Korzystanie z przewagi nad przeglądarką i możliwości kontrolowania renderowanej strony, możesz stworzyć środowisko, które może zachęcić użytkownika do wprowadzenia kompromitujących informacji, aby można je było przechwycić i wykorzystać. Możesz nakłonić użytkownika do nieświadomego przyznania uprawnień do wystąpienia w inny sposób zabezpieczonego zdarzenia, takiego jak uruchomienie dowolnego programu lub przyznanie dostępu do zasobów lokalnych. Możesz tworzyć ukryte okna dialogowe i przezroczyste ramki lub kontrolować zdarzenia myszy, aby pomóc w tym celu, zaprzeczając prawdziwej funkcji interfejsu użytkownika i przedstawiając fałszywe wrażenie użytkownikowi.

Atakowanie przeglądarek

Kategoria Atakowanie przeglądarek obejmuje bezpośrednie ataki na samą podstawową przeglądarkę. Zatapiasz się w tym w Części 6, w którym badasz różne obszary, od dostawców odcisków palców po pełne wykorzystanie. Przeglądarka internetowa to mamut powierzchni ataku. Istnieje szeroka gama interfejsów API i abstrakcji do przechowywania i przywoływania danych. Nic dziwnego, że przeglądarki internetowe od lat są nękane lukami w takiej czy innej formie. Co bardziej zaskakujące, twórcy przeglądarki internetowej robią to dobrze tyle razy, co oni. Poznasz różne klasy luk rozszerzeń. Luki w rozszerzeniach mogą być wykorzystywane do wykorzystywania funkcji znajdujących się w nich do przeprowadzania żądań między źródłami, a nawet wykonywania poleceń systemu operacyjnego.

Atakowanie wtyczek

Jednym z najbardziej podatnych na ataki obszarów przeglądarki internetowej są wtyczki. Wtyczka różni się znacznie od rozszerzenia, ponieważ są komponentami stron trzecich, które są inicjowane wyłącznie według uznania obsługiwanej strony internetowej (w przeciwieństwie do stałego włączania do przeglądarki). Kategoria Atakowanie wtyczek w metodologii została omówiona w rozdziale 8. Obejmuje to ataki na wszechobecne wtyczki, takie jak Java i Flash. Odkrywasz jak odkryć, jakie wtyczki są zainstalowane, ujawnić możliwe do wykorzystania historyczne słabości wykryte przez różnych badaczy w tej dziedzinie i dowiedzieć się, jak można ominąć niektóre funkcje bezpieczeństwa zaprojektowane w celu ochrony przed nadużywaniem wtyczek.

Atakowanie aplikacji internetowych

Przeglądarka jest zbudowana do korzystania z sieci, więc nie powinno dziwić, że w metodologii istnieje faza Atakowanie aplikacji internetowych. Obszar ten obejmuje atakowanie aplikacji internetowych przy użyciu standardowej funkcjonalności przeglądarki internetowej. Część 9 zagłębia się w wykorzystanie standardowej funkcjonalności przeglądarki do wykorzystania aplikacji internetowych. Wyobraź sobie bogactwo powszechnie dostępnych aplikacji dostępnych w intranecie tylko z obszaru organizacji. Co się stanie, jeśli zewnętrzna witryna w innej zakładce może przeglądać te witryny? Dowiesz się, że założenie, że witryny intranetowe są chronione przed zewnętrznymi atakami przez zaporę sieciową, jest ewidentnie fałszywe.

Atakowanie sieci

Być może nie zauważyłeś, że twoja przeglądarka internetowa łączy się z niestandardowymi portami, ale ten scenariusz jest dość powszechny. Aplikacje często instalują serwer WWW na dowolnym porcie, a niektóre witryny w Internecie publikują nawet swoje treści na portach innych niż 80 lub 443. Co by było, gdyby Twoja przeglądarka w ogóle nie łączyła się z serwerem WWW? Co by było, gdyby łączyło się z usługą, która spełnia zupełnie inny cel i używa zupełnie innego protokołu? Nie naruszałoby to SOP i w większości przypadków byłoby ważne z punktu widzenia zabezpieczeń przeglądarki. Zmiana przeznaczenia tych zachowań przeglądarki pozwala na realizację skomplikowanych scenariuszy ataków. Faza atakowania sieci przeskakuje do atakowania niższych warstw modelu OSI. W rozdziale 10 odkryjesz, że wszystkie techniki można w równym stopniu zastosować do atakowania dowolnej sieci TCP/IP.

Podsumowanie

Prawdopodobnie przeglądarka internetowa jest najważniejszym oprogramowaniem tej dekady. Dostawcy oprogramowania rzadko opracowują niestandardowe oprogramowanie klienckie dla swoich aplikacji. Częściej opracowują interfejsy użytkownika aplikacji za pomocą technologii internetowej; nie tylko tradycyjne internetowe aplikacje internetowe, ale także aplikacje lokalne i intranetowe. Przeglądarka internetowa dominuje na pozycji klienta w modelu serwer-klient. Przeglądarka internetowa ma już władzę w prawie wszystkich sieciach i nawet gdyby chcieli usunąć ją z jakiejkolwiek organizacji, jest mało prawdopodobne, aby udało się to osiągnąć. Organizacja nie ma wyboru i musi mieć w swojej sieci przeglądarki internetowe.

Hakerzy zazwyczaj atakują podszywając się pod niezłośliwy serwer sieciowy wysyłający prawidłową komunikację do przeglądarki internetowej. W większości przypadków przeglądarka internetowa nie będzie wiedziała, że komunikuje się z fałszywym serwerem sieciowym. Przeglądarka wykonuje wszystkie instrukcje wysyłane przez fałszywy serwer sieciowy w rzekomo bezpiecznym schronieniu wewnątrz zapory sieciowej. W pozostałej części tej książki opanujesz metodologię i poznasz techniki wykorzystywania przeglądarki internetowej i urządzeń, do których ma ona dostęp.

Pytanie

1. Jaką funkcję ma DOM w przeglądarce internetowej?

2. Dlaczego posiadanie bezpiecznej przeglądarki jest ważne dla całościowego podejścia do bezpieczeństwa?

3. Wymień niektóre różnice między JavaScript i VBScript.

4. Wymień trzy sposoby, w jakie serwer może zmniejszyć bezpieczeństwo przeglądarki internetowej.

5. Jaka jest powierzchnia ataku przeglądarki internetowej?

6. Opisz piaskownicę.

7. Kiedy przeglądarka używa protokołu HTTPS do komunikacji, czy serwer proxy może wyświetlić komunikację?

8. Nazwij trzy nagłówki HTTP związane z bezpieczeństwem.

9. Dlaczego zasada solidności nie jest przyjacielem pracownika ochrony?

10. Jaki język skryptowy jest dostępny w Internet Explorerze, a nie w innych nowoczesnych przeglądarkach?