Wstrzyknięcia SQL i przepełnienie bufora to techniki hakerskie wykorzystywane do wykrywania słabości stron aplikacji. Gdy programy są napisane, niektóre parametry użyte w tworzeniu kodu aplikacji mogą pozostawiać słabości w programie. Wstrzyknięcia SQL i przepełnienia bufora są uwzględnione w tej samej części, ponieważ obie są metodami używanymi do atakowania aplikacji i są generalnie powodowane przez błędy programistyczne. Zasadniczo celem iniekcji SQL jest przekonanie aplikacji do uruchomienia kodu SQL, który nie był przeznaczony dla niej . SQL injection to metoda hakerska używana do atakowania baz danych SQL, a przepełnienia buforów mogą występować w wielu różnych typach aplikacji. Wstrzyknięcia SQL i przepełnienia bufora są podobnymi exploitami, ponieważ oba są zwykle dostarczane za pośrednictwem pola wejściowego użytkownika. W polu wejściowym użytkownik może wprowadzić nazwę użytkownika i hasło na stronie internetowej, dodać dane do adresu URL lub wyszukać słowo kluczowe w innej aplikacji. Luka SQL injection jest spowodowana głównie przez niezweryfikowane lub nieodkażone dane wprowadzane przez użytkownika za pośrednictwem tych pól. Zarówno luki w zabezpieczeniach SQL Server, jak i przepełnienie bufora są spowodowane tym samym problemem: niepoprawne parametry, które nie zostały zweryfikowane przez aplikację. Jeśli programiści nie poświęcą czasu na sprawdzenie zmiennych, które użytkownik może wprowadzić w polu zmiennej, wyniki mogą być poważne i nieprzewidywalne. Wyrafinowani hakerzy mogą wykorzystać tę lukę, powodując błąd wykonania i zamknięcie systemu lub aplikacji lub powłokę poleceń, która zostanie wykonana dla hakera. Instancje do iniekcji SQL i zabezpieczenia przed przepełnieniem bufora zostały zaprojektowane w celu wykorzystania bezpiecznych metod programowania. Zmieniając zmienne używane przez kod aplikacji, można w znacznym stopniu ograniczyć słabości aplikacji. W tym rozdziale szczegółowo opisano, jak wykonać iniekcję SQL i atak przepełnienia bufora i zbadać najlepsze środki zaradcze, aby zapobiec atakowi.

Injection SQL

Jako etyczny hacker ważne jest, abyś mógł zdefiniować iniekcje SQL i zrozumieć kroki, które haker podejmuje, by przeprowadzić atak SQL injection. Ponadto powinieneś znać luki w zabezpieczeniach SQL Server, a także środki zaradcze do ataków typu SQL injection. Wstrzyknięcie SQL występuje, gdy aplikacja przetwarza dane dostarczone przez użytkownika w celu utworzenia instrukcji SQL bez sprawdzania poprawności danych wejściowych. Dane użytkownika są następnie przesyłane do serwera baz danych aplikacji WWW w celu wykonania. Po pomyślnym wykorzystaniu iniekcja SQL może dać atakującemu dostęp do zawartości bazy danych lub zezwolić hakerowi na zdalne wykonywanie poleceń systemowych. W najgorszym przypadku haker może przejąć kontrolę nad serwerem hostującym bazę danych. Ten exploit może zapewnić hakerowi dostęp do zdalnej powłoki w systemie plików serwera. Wpływ ataków typu SQL injection zależy od tego, gdzie luka jest w kodzie, jak łatwo jest wykorzystać lukę w zabezpieczeniach i jaki jest poziom dostępu aplikacji do bazy danych. Teoretycznie iniekcja SQL może występować w dowolnym typie aplikacji, ale najczęściej jest związana z aplikacjami internetowymi, ponieważ najczęściej są atakowane. Jak wcześniej wspomniano ,aplikacje internetowe są łatwym celem, ponieważ z samej swojej natury są otwarte na dostęp do nich z Internetu. Powinieneś mieć podstawową wiedzę na temat działania baz danych i używania poleceń SQL w celu uzyskania dostępu do informacji w bazach danych przed przystąpieniem do egzaminu CEH. Podczas ataku typu SQL injection aplikacja internetowa wprowadza złośliwy kod do pola formularza internetowego lub kodu strony internetowej, aby system wykonał powłokę poleceń lub inne dowolne polecenia. Tak jak prawowity użytkownik wprowadza zapytania i dodatki do bazy danych SQL za pośrednictwem formularza internetowego, haker może wstawiać polecenia do SQL Server za pośrednictwem tego samego formularza internetowego. Na przykład dowolne polecenie hakera może otworzyć wiersz polecenia lub wyświetlić tabelę z bazy danych. Tabela bazy danych może zawierać dane osobowe, takie jak numery kart kredytowych, numery ubezpieczenia społecznego lub hasła. Serwery SQL są bardzo popularnymi serwerami baz danych i używane przez wiele organizacji do przechowywania poufnych

danych. Dzięki temu serwer SQL jest celem o wysokiej wartości, a zatem bardzo atrakcyjnym dla hakerów.

Wykrywanie luk SQL Injection

Podczas przeprowadzania testu penetracji czarnego kapelusza w sieci korporacyjnej, tester bezpieczeństwa Tom znalazł niestandardową aplikację na jednym z publicznie dostępnych serwerów internetowych. Ponieważ był to test black-hat, Tom nie miał dostępu do kodu źródłowego, aby zobaczyć, jak program został stworzony. Ale po przeprowadzeniu gromadzenia informacji udało mu się ustalić, że na serwerze działał Microsoft Internet Information Server 6 wraz z ASP.NET, co sugerowało, że bazą danych był Microsoft SQL Server. Strona logowania do aplikacji internetowej zawierała nazwę użytkownika, pole hasła i zapomniane łącze do hasła, co stanowiło najłatwiejszą drogę do systemu. Zapomniane połączenie z hasłem działa, sprawdzając w bazie danych użytkownika adres e-mail użytkownika i wysyłając wiadomość e-mail zawierającą hasło do tego adresu. Aby ustalić, czy link do zapomnianego hasła był podatny na wstrzyknięcie SQL, Tom wprowadził pojedynczy cudzysłów jako część danych w polu zapomnianego hasła. Celem było sprawdzenie, czy aplikacja mogłaby skonstruować ciąg znaków SQL dosłownie bez odkażania danych wprowadzanych przez użytkownika. Przesyłając formularz z cudzysłowem na adres e-mail, otrzymał błąd 500 (awaria serwera), a to sugerowało, że dane wejściowe użytkownika były przetwarzane dosłownie. Podstawowy kod SQL formularza prawdopodobnie wyglądał mniej więcej tak:

SELECT fieldlist

FROM table

WHERE field = ‘$EMAIL’;

Tom wpisał swój adres e-mail, a następnie pojedynczy cudysłów w polu zapomnianego adresu e-mail. Analizator składni SQL aplikacji internetowej znalazł dodatkowy znacznik cudzysłowu i został przerwany z błędem składni. Kiedy Tom otrzymał ten komunikat o błędzie, był w stanie stwierdzić, że dane wejściowe użytkownika nie zostały odpowiednio oczyszczone i że aplikacja może zostać wykorzystana. W tym przypadku nie musiał kontynuować i wykorzystywać aplikacji, ponieważ komunikat o błędzie był wystarczającym dowodem na to, że aplikacja była podatna na atak typu SQL injection. W wyniku tego testu penetracji klient był w stanie naprawić lukę w SQL Server.

Znalezienie luki SQL Injection

Przed uruchomieniem ataku typu SQL injection haker określa, czy konfiguracja bazy danych oraz powiązanych tabel i zmiennych jest podatna na ataki. Kroki w celu ustalenia luki w zabezpieczeniach programu SQL Server są następujące:

1. Za pomocą przeglądarki internetowej wyszukaj stronę internetową, która korzysta ze strony logowania lub innych pól danych wejściowych bazy danych lub zapytań (takich jak formularz "Nie pamiętam hasła"). Poszukaj stron internetowych wyświetlających polecenia POST lub GET HTML, sprawdzając kod źródłowy witryny.

2. Przetestuj serwer SQL za pomocą pojedynczych cudzysłowów (''). Ta operacja wskazuje, czy zmienna wejściowa użytkownika jest odkażana lub interpretowana dosłownie przez serwer. Jeśli serwer odpowie komunikatem o błędzie z informacją "a" = "a" (lub czymś podobnym), najprawdopodobniej jest podatny na atak typu SQL injection.

3. Użyj polecenia SELECT, aby pobrać dane z bazy danych lub polecenie INSERT, aby dodać informacje do bazy danych. Oto kilka przykładów tekstu zmiennej pól, które można wykorzystać w formularzu internetowym do testowania luk SQL:

* Blah 'lub 1 = 1-

* Zaloguj się: blah 'lub 1 = 1-

* Hasło :: bla "lub 1 = 1-

* http: //search/index.asp? Id = blah 'lub 1 = 1-

Te polecenia i podobne odmiany mogą pozwolić użytkownikowi na ominięcie logowania w zależności od struktury bazy danych. Po wprowadzeniu w polu formularza polecenia mogą zwrócić wiele wierszy w tabeli lub nawet całą tabelę bazy danych, ponieważ SQL Server interpretuje je dosłownie. Podwójne myślniki pod koniec polecenia informują SQL, aby zignorował resztę polecenia jako komentarz.

Oto kilka przykładów użycia poleceń SQL do przejęcia kontroli. Aby uzyskać listing katalogu, wpisz następujące polecenie w polu formularza:

Blah '; exec master..xp_cmdshell "dir c: \ *. * / S> c: \ katalog.txt" -

Aby utworzyć plik, wpisz następujące informacje w polu formularza:

Blah '; exec master..xp_cmdshell "echo hacker-was-here> c: \ hacker.txt" -

Aby pingować adres IP, wpisz następujące polecenie w polu formularza:

Blah '; exec master..xp_cmdshell "ping 192.168.1.1" -

Cele SQL Injection

Ataki typu SQL injection są wykorzystywane przez hakerów do osiągnięcia określonych rezultatów. Niektóre exploity SQL odtwarzają cenne dane użytkowników przechowywane w bazie danych, a niektóre są tylko prekursorami innych ataków. Oto najczęstsze cele ataku typu SQL injection:

Identyfikacja luki w zabezpieczeniach SQL Injection Celem jest zbadanie aplikacji internetowej, aby dowiedzieć się, które parametry i pola wejściowe użytkownika są podatne na iniekcję SQL.

Wykonywanie fingerprintingu bazy danych Celem jest wykrycie typu i wersji bazy danych, z której korzysta aplikacja internetowa, oraz "pobranie odcisków palców" bazy danych. Znajomość typu i wersji bazy danych wykorzystywanej przez aplikację internetową umożliwia atakującemu spreparowanie ataków związanych z bazami danych.

Określanie schematu bazy danych Aby poprawnie wyodrębnić dane z bazy danych, osoba atakująca często musi znać informacje o schemacie bazy danych, takie jak nazwy tabel, nazwy kolumn i typy danych kolumn. Ta informacja może zostać wykorzystana w kolejnym ataku.

Wyodrębnianie danych Tego typu ataki wykorzystują techniki, które będą wyodrębniać wartości danych z bazy danych. W zależności od rodzaju aplikacji internetowej informacje te mogą być bardzo czułe i wysoce pożądane dla atakującego.

Dodawanie lub modyfikowanie danych Celem jest dodanie lub zmiana informacji w bazie danych.

Wykonywanie ataku Denial of Service Ataki te są przeprowadzane w celu wyłączenia dostępu do aplikacji WWW, a tym samym odmowy usługi innym użytkownikom. Ataki polegające na blokowaniu lub upuszczaniu tabel bazy danych również należą do tej kategorii.

Unikanie wykrywania Ta kategoria odnosi się do pewnych technik ataku, które są stosowane w celu uniknięcia audytu i wykrywania.

Pomijanie uwierzytelniania Celem jest umożliwienie atakującemu pominięcia mechanizmów uwierzytelniania baz danych i aplikacji. Pomijanie takich mechanizmów może umożliwić atakującemu przejęcie praw i przywilejów związanych z innym użytkownikiem aplikacji.

Wykonywanie zdalnych poleceń Tego typu ataki próbują wykonywać dowolne polecenia w bazie danych. Te polecenia mogą być procedurami przechowywanymi lub funkcjami dostępnymi dla użytkowników baz danych.

Przeprowadzanie eskalacji uprawnień Te ataki wykorzystują błędy implementacji lub błędy logiczne w bazie danych w celu eskalacji uprawnień atakującego.

Injection SQL za pomocą ciągów dynamicznych

Większość aplikacji SQL wykonuje określone, przewidywalne zadanie. Wiele funkcji bazy danych SQL otrzymuje statyczne dane wejściowe użytkownika, gdzie jedyną zmienną są pola wprowadzania użytkownika. Takie instrukcje nie zmieniają się z wykonania do wykonania. Są one powszechnie nazywane statycznymi instrukcjami SQL. Jednak niektóre programy muszą budować i przetwarzać różne instrukcje SQL w środowisku wykonawczym. W wielu przypadkach pełny tekst instrukcji jest nieznany do momentu wykonania wniosku. Takie instrukcje mogą i prawdopodobnie będą zmieniać się z realizacji na wykonanie. Są więc nazywane dynamicznymi instrukcjami SQL. Dynamiczny SQL to rozszerzona forma SQL, która w przeciwieństwie do standardowego SQL ułatwia automatyczne generowanie i wykonywanie instrukcji programu. Dynamiczny SQL to termin używany do oznaczania kodu SQL, który jest generowany przez aplikację internetową przed jej wykonaniem. Dynamic SQL to elastyczne i wydajne narzędzie do tworzenia łańcuchów SQL. Może to być pomocne, gdy okaże się, że konieczne jest napisanie kodu, który można dostosować do różnych baz danych, warunków lub serwerów. Dynamiczny SQL ułatwia również automatyzację zadań powtarzanych wielokrotnie w aplikacji internetowej. Haker może zaatakować internetową formę uwierzytelniania za pomocą iniekcji SQL za pomocą dynamicznych ciągów. Na przykład podstawowy kod formularza uwierzytelniania internetowego na serwerze WWW może wyglądać następująco:

SQLCommand = “SELECT Username FROM Users WHERE Username = ‘“

SQLCommand = SQLComand & strUsername

SQLCommand = SQLComand & “‘ AND Password = ‘“

SQLCommand = SQLComand & strPassword

SQLCommand = SQLComand & “‘“

strAuthCheck = GetQueryResult(SQLQuery)

Haker może wykorzystać lukę SQL injection, wprowadzając login i hasło w formularzu internetowym, który wykorzystuje następujące zmienne:

Username: kimberly

Password: graves’ OR ‘’=’

Aplikacja SQL utworzy ciąg poleceń z tego wejścia w następujący sposób:

SELECT Username FROM Users

WHERE Username = ‘kimberly’

AND Password = ‘graves’ OR ‘’=’’

Jest to przykład iniekcji SQL: to zapytanie zwróci wszystkie wiersze z bazy danych użytkownika, niezależnie od tego, czy kimberly jest prawdziwą nazwą użytkownika w bazie danych, czy groby są uzasadnionym hasłem. Wynika to z instrukcji OR dołączonej do klauzuli WHERE. Porównanie '' = '' zawsze zwróci prawdziwy wynik, dzięki czemu ogólna klauzula WHERE będzie miała wartość true dla wszystkich wierszy w tabeli. Umożliwi to hakerowi zalogowanie się za pomocą dowolnej nazwy użytkownika i hasła. W ćwiczeniu 9.1 użyjesz HP Scramlr do testowania podatności na iniekcje SQL.

Ćwiczenie 9 .1

Korzystanie z narzędzia Scrawlr firmy HP do testowania słabych punktów SQL Injection

1. Pobierz Scrawlr ze strony www.HP.com.

2. Zainstaluj Scrawlr na swoim komputerze z systemem Windows.

3. Otwórz program Scrawlr.

4. Wpisz docelowy adres internetowy w polu Adres URL strony do skanowania:

5. Kliknij przycisk Start, aby rozpocząć kontrolę witryny pod kątem luk SQL injection.

6. Po zakończeniu skanowania luki SQL injection Scrawlr wyświetli dodatkowe hosty połączone ze skanowaną stroną. Najlepszą metodą jest skanowanie połączonych stron, a także witryny głównej, aby upewnić się, że nie występują luki w zabezpieczeniach SQL Injection.

SQL Injection .Środki zaradcze

Przyczyna luk SQL injection jest stosunkowo prosta i dobrze zrozumiała: niewystarczająca walidacja danych wprowadzanych przez użytkownika. Aby rozwiązać ten problem, podczas programowania aplikacji można używać defensywnych praktyk kodowania, takich jak kodowanie danych wprowadzanych przez użytkownika i sprawdzanie poprawności. Jest to pracochłonny i czasochłonny proces sprawdzania wszystkich aplikacji pod kątem luk SQL injection. Podczas implementacji środków przeciwdziałających iniekcjom SQL, sprawdź kod źródłowy pod kątem następujących słabości programistycznych:

* Pojedyncze cytaty

* Brak sprawdzenia poprawności danych wejściowych

Pierwszy środek zaradczy w zapobieganiu atakowi typu SQL injection minimalizuje przywileje połączenia użytkownika z bazą danych i wymuszanie silnych haseł dla kont SA i Administratora. Powinieneś również wyłączyć pełne komunikaty o błędach lub objaśnieniach, aby haker nie otrzymał więcej informacji niż jest to konieczne; takie informacje mogą pomóc w ustaleniu, czy serwer SQL jest podatny na atak. Pamiętaj, że jednym z celów iniekcji SQL jest uzyskanie dodatkowych informacji o tym, które parametry są podatne na atak. Kolejnym środkiem zapobiegającym iniekcji SQL jest sprawdzanie danych wprowadzanych przez użytkownika i sprawdzanie poprawności danych przed wysłaniem danych wejściowych do aplikacji w celu przetworzenia. Niektóre środki zaradcze do iniekcji SQL to

* Odrzucanie znanych błędnych danych wejściowych

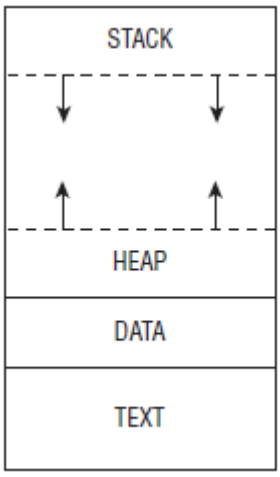
* Odkażanie i sprawdzanie poprawności pola wejściowego

Przepełnienie bufora

Jako etyczny hacker musisz być w stanie zidentyfikować różne typy przepełnień bufora. Powinieneś również wiedzieć, jak wykryć lukę w zabezpieczeniach polegającą na przepełnieniu bufora i zrozumieć kroki, które haker może wykorzystać do wykonania ataku opartego na stosie. Przyjrzymy się tym zagadnieniom, a także omówimy techniki mutacji przepełnienia bufora w poniższych sekcjach.

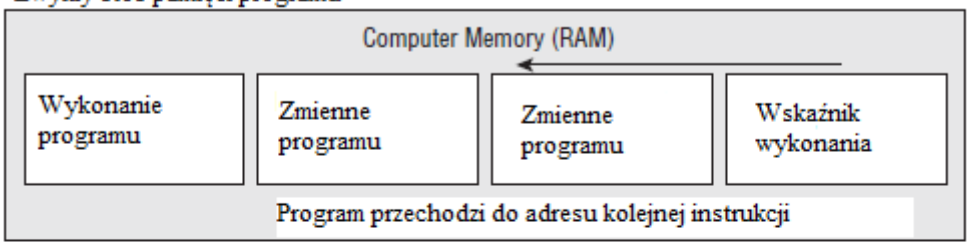
Rodzaje przepełnień bufora i metody wykrywania

Przepełnienie bufora to exploity wykorzystywane przez hakerów przeciwko systemowi operacyjnemu lub aplikacji; podobnie jak ataki typu SQL injection, zwykle są kierowane na pola wprowadzania danych przez użytkownika. Exploit przepełnienia bufora powoduje awarię systemu przez przeciążenie pamięci lub wykonanie powłoki poleceń lub dowolnego kodu w systemie docelowym. Luka związana z przepełnieniem bufora jest spowodowana brakiem sprawdzania ograniczeń lub brakiem sanityzacji sprawdzania poprawności danych wejściowych w zmiennym polu (na przykład w formularzu internetowym). Jeśli aplikacja nie sprawdzi ani nie sprawdzi rozmiaru lub formatu zmiennej przed wysłaniem jej do pamięci, zostanie usunięta luka w zabezpieczeniach. Dwa typy przepełnień bufora są oparte na stosach i oparte na stertach. Stos i sterty są miejscami przechowywania zmiennych dostarczanych przez użytkownika w uruchomionym programie. Zmienne są przechowywane w stosie lub stercie, dopóki program ich nie potrzebuje. Stosy są statycznymi lokalizacjami przestrzeni adresowej pamięci, podczas gdy stosy są dynamicznymi przestrzeniami adresowymi pamięci, które występują podczas działania programu. Przepełnienie bufora oparte na sterty występuje w dolnej części pamięci i nadpisuje inne zmienne dynamiczne. Patrz rysunek

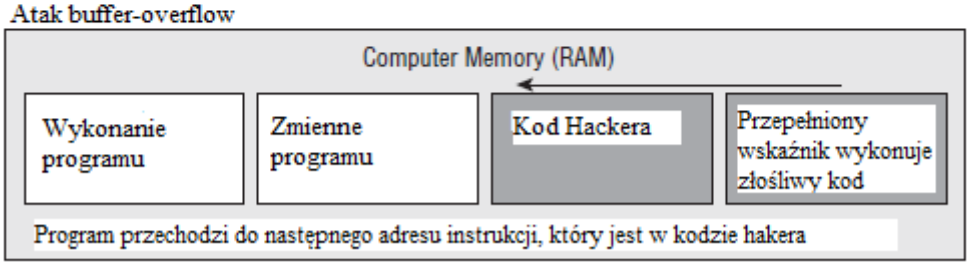


Stos wywołań lub stos jest używany do śledzenia miejsca, w którym w kodzie programowania wskaźnik wykonywania powinien powracać po wykonaniu każdego fragmentu kodu. Atak polegający na przepełnieniu bufora na stosie

Zwykły stos pamięci programu



Atak buffer-overflow



występuje, gdy pamięć przypisana do każdej procedury wykonywania jest przepełniona. W wyniku obu typów przepełnień bufora program może otworzyć powłokę lub wiersz polecenia lub zatrzymać wykonywanie programu. W następnej sekcji opisano ataki typu przepełnienie bufora oparte na stosie. Aby wykryć luki w przepełnieniu bufora programu, które wynikają z nieprawidłowo napisanego źródła kod, haker wysyła duże ilości danych do aplikacji za pośrednictwem pola formularza i widzi, co robi program. Poniżej przedstawiono kroki, które haker wykorzystuje do przepełnienia bufora stosowego:

1. Wprowadź zmienną do bufora, aby wyczerpać ilość pamięci w stosie.

2. Wprowadź więcej danych niż bufor został przydzielony w pamięci dla tej zmiennej, która powoduje przepełnienie pamięci lub uciekanie w miejsce w pamięci na następny proces. Następnie dodaj kolejną zmienną i nadpisz wskaźnik powrotu, który mówi programowi gdzie powrócić do po wykonaniu zmiennej.

3. Program wykonuje tę złośliwą zmienną kodu, a następnie używa wskaźnika powrotu, aby powrócić do następnej linii kodu wykonywalnego. Jeśli haker z powodzeniem zastąpi wskaźnik, program wykona kod hakera zamiast kodu programu.

Większość hakerów nie musi być zaznajomiona ze szczegółami przepełnień bufora. Presety exploitów można znaleźć w Internecie i są wymieniane między grupami hakerskimi. Ćwiczenie 9.2 pokazuje użycie Metasploit do wykonania ataku przepełnienia bufora.

Rejestr pamięci, który zostanie nadpisany adresem zwrotnym kodu exploita jest znany jako EIP.

Ćwiczenie 9. 2

Przeprowadzanie ataku z przepełnieniem bufora przy użyciu Metasploit

1. Otwórz Metasploit Framework.

2. Uruchom maszynę testową z systemem Windows Server z IIS.

3. Z Metasploit przeprowadź atak przepełnienia bufora IIS na maszynie testowej z IIS.

4. Wybierz ładunek do dostarczenia do systemu docelowego IIS za pomocą exploita przepełnienie bufora.

Przeciwdziałanie przepełnieniu bufora

Jak widać, hakerzy mogą ukończyć standardową procedurę przepełnienia bufora, aby przekierować wskaźnik powrotu do kodu, który wybrał. Haker musi znać dokładny adres pamięci i rozmiar stosu, aby wskaźnik powrotu wykonał swój kod. Haker może użyć instrukcji No Operation (NOP), która jest po prostu dopełnieniem, aby przesunąć wskaźnik instrukcji i nie wykonuje żadnego kodu. Instrukcja NOP jest dodawana do ciągu znaków przed wykonaniem złośliwego kodu. Jeśli system wykrywania włamań (IDS) jest obecny w sieci, może udaremnić próby hakera, który wysyła serię instrukcji NOP do przekazania do wskaźnika instrukcji. Aby ominąć IDS, haker może losowo zastąpić niektóre instrukcje NOP odpowiednimi fragmentami kodu, takimi jak x ++, x - ;? NOPNOP. Ten przykład zmutowanego ataku przepełnienia bufora może pomiń wykrywanie przez IDS. Programiści nie powinni używać wbudowanych funkcji strcpy (), strcat () i streadd () C / C ++, ponieważ są one podatne na przepełnienie bufora. Ewentualnie język Java może być używany jako język programowania, ponieważ Java nie jest podatna na przepełnienie bufora.

Podsumowanie

Wstrzyknięcia SQL i przepełnienie bufora to metody hakerskie wykorzystywane do wykorzystywania aplikacji. Aplikacje internetowe są szczególnie podatne na ataki, ponieważ mają łatwy dostęp dla hakerów w postaci pól wejściowych użytkownika, takich jak nazwa użytkownika, hasło, zapomniane hasło i pola cen. Ścisła interpretacja i nieodkażone dane wejściowe mogą bezpośrednio wchodzić w interakcję z bazą danych i mogą powodować, że baza danych ujawnia poufne informacje. Przepełnienia buforów występują w dwóch typach, opartych na stosach i opartych na sterty, które atakują różne obszary przestrzeni alokacji pamięci. Ataki typu SQL injection i przepełnienie bufora można zapobiec, sprawdzając poprawność danych wprowadzanych przez użytkownika i ograniczając długość pola wejściowego użytkownika. Te dwa środki zaradcze mogą naprawić większość luk w aplikacjach i chronić aplikacje przed atakami SQL Injection i przepełnieniami bufora.

Do Zapamiętania !

* Dowiedziałeś się, jak ataki typu SQL injection i przepełnienia bufora są podobne. Wstrzyknięcia SQL i przepełnienia bufora są podobne, ponieważ oba ataki są dostarczane za pośrednictwem formularza internetowego.

* Zapoznałeś się z celami SQL injection. Celem ataków typu SQL injection może być uzyskanie danych użytkownika z bazy danych lub zbieranie informacji o lukach w bazach danych i aplikacjach.

* Zapoznałeś się z przeciwdziałaniem iniekcjom SQL. Wykorzystanie poprawnego kodu programowania bez pojedynczych cudzysłowów oraz sprawdzanie granic i sprawdzanie danych wejściowych to środki zaradcze SQL injection.

* Poznałeś różnicę między przepełnieniem bufora opartym na sterty i stertą. Stosy są statycznymi lokalizacjami przestrzeni adresowej pamięci, podczas gdy stosy są dynamicznymi przestrzeniami adresowymi pamięci.

* Dowiedziałeś się, jak ominąć IDS za pomocą ataku polegającego na przepełnieniu bufora. IDS szuka serii instrukcji NOP. Zastępując instrukcję NOP innymi segmentami kodu, haker może skutecznie ominąć IDS.

* Zapoznałeś się z przepełnieniem bufora i przeciwdziałaniem iniekcjom SQL. Sprawdzanie granic i odkażanie danych wejściowych z formularza internetowego może zapobiegać przepełnieniu bufora i podatności na iniekcje SQL