Najważniejsze kroki, które należy podjąć, aby serwer Apache był bardziej bezpieczny

Przegląd

Apache to najpopularniejszy serwer WWW typu open source dostępny dla nowoczesnych serwerów Linux. Oferuje elastyczną konfigurację pozwalającą na szeroki zakres zastosowań, od obsługi podstawowych stron HTML, przez złożone aplikacje PHP/Passenger, po pośredniczenie w żądaniach jako brama odwrotnego proxy. Biorąc pod uwagę jego popularność i łatwość użytkowania, niezbędne jest zainstalowanie i utrzymywanie bezpiecznego środowiska dla instalacji Apache.

W tym artykule założono, że zainstalowałeś i skonfigurowałeś Apache w instancji Debiana lub Ubuntu:

- Jak skonfigurować Apache na DreamCompute z systemem Debian lub Ubuntu

> Ten artykuł dotyczy tylko serwerów dedykowanych z użytkownikiem Admin lub DreamCompute, ponieważ tylko te plany zezwalają na uprawnienia sudo.

Aktualizuj Apache

Apache ma dobre wyniki w zakresie bezpieczeństwa, a błędy bezpieczeństwa rzadko występują w samym serwerze WWW. Mimo to ważne jest, aby aktualizować Apache, aby korzystać z najnowszych dostępnych zabezpieczeń, stabilności i funkcji. Zasadniczo jest to po prostu kwestia aktualizowania pakietu Apache dostarczanego przez system operacyjny danej dystrybucji (np. przez *apt* , *yum* , itp.). Zaleca się również, aby operatorzy serwerów Apache obserwowali listę mailingową Apache Server Announcements , aby być na bieżąco z najnowszymi wiadomościami od zespołu programistów Apache. Możesz zapisać się na listę tutaj:

- Listy mailingowe serwera Apache HTTP

Zabezpieczanie konfiguracji

Apache jest zbudowany tak, aby był stabilny i bezpieczny, ale będzie tak bezpieczny, jak użytkownik, który go konfiguruje. Po zbudowaniu i zainstalowaniu Apache ważne jest, aby skonfigurować serwer tak, aby był jak najmniejszy.

Uruchom jako użytkownik nieuprzywilejowany

W bezpieczeństwie zasada najmniejszych uprawnień mówi, że jednostka nie powinna mieć więcej uprawnień niż jest to konieczne do osiągnięcia jej celów w danym systemie. W kontekście twojego serwera WWW oznacza to zablokowanie Apache, aby działał tylko z uprawnieniami niezbędnymi do uruchomienia. Pierwszym krokiem w tym procesie jest skonfigurowanie Apache tak, aby działał jako nieuprzywilejowany użytkownik systemu (np. nie root). Odbywa się to poprzez ustawienie zmiennych `APACHE_RUN_USER` i `APACHE_RUN_GROUP` w **pliku /etc/apache2/envvars** :

Edytuj plik i zmień następujące wiersze:

```
[użytkownik@instancja]$ sudo vim /etc/apache2/envvars
# Ponieważ nie ma rozsądnego sposobu na uzyskanie przeanalizowanej konf
# ustawienia są definiowane za pomocą zmiennych środowiskowych, a nastę
# /etc/init.d/apache2, /etc/logrotate.d/apache2 itp.
eksportuj APACHE_RUN_USER=apache
eksportuj APACHE_RUN_GROUP=apache
```

Serwery Apache dystrybuowane jako wspólny pakiet systemu operacyjnego mogą również używać nazwy użytkownika i grupy, takiej jak **www-data** lub **Nobody** . Niezależnie od wyboru nazwy użytkownika upewnij się, że wybrany **użytkownik/grupa** ma tak mało uprawnień, jak to konieczne do prawidłowego działania.

Wyłącz tokeny serwera

Specyfikacja HTTP zaleca (ale nie wymaga), aby serwery WWW identyfikowały się za pomocą nagłówka `serwera .` W przeszłości serwery WWW zawierały informacje o wersji jako część tego nagłówka. Ujawnianie wersji działającego Apache może być niepożądane, szczególnie w środowiskach wrażliwych na ujawnianie informacji. Skonfiguruj Apache, aby nie wyświetlał swojej wersji w nagłówku `serwera` , edytując następujący plik:

```
[użytkownik@instancja]$ sudo vim /etc/apache2/conf-available/security.c
```

W tym pliku zobaczysz kilka ServerTokens . Upewnij się, że wszystkie są skomentowane i pojawia się tylko `ProductOnly` .

```
#Tokeny serwera
#ServerTokens Minimalne
System operacyjny #ServerTokens
#ServerTokens pełne
Tylko produkt ServerTokens;
```

Uruchom ponownie Apache, aby zaktualizować zmiany.

```
[użytkownik@instancja]$ Sudo przeładuj usługę Apache2
```

Wyłącz pliki .htaccess

Pliki .htaccess to potężna funkcja, która pozwala Apache na rozszerzenie konfiguracji poza główny plik konfiguracyjny. Chociaż może to być wygodne, stanowi zagrożenie dla bezpieczeństwa, ponieważ Apache odczyta każdy plik **.htaccess** na swojej ścieżce — nawet te umieszczone przez atakującego, który mógłby zagrozić serwerowi. Pożądane może być zablokowanie konfiguracji Apache poprzez całkowite wyłączenie plików **.htaccess .** Można to zrobić, edytując dyrektywę `AllowOverride` **w pliku /etc/apache2/apache2.conf:**

```
[użytkownik@instancja]$ sudo vim /etc/apache2/apache2.conf Zezwól na
zastępowanie Brak
```

Dodatkowo, szczegółowa kontrola nad tym, które dyrektywy Apache mogą być używane w plikach **.htaccess** , może być również kontrolowana przez `AllowOverride` :

```
Zezwalaj na zastępowanie indeksów AuthConfig
```

W powyższym przykładzie wszystkie dyrektywy, które nie należą ani do grupy `AuthConfig` , ani do grupy Indexes, powodują wewnętrzny błąd serwera. Zobacz następującą witrynę, aby uzyskać więcej informacji:

- https://httpd.apache.org/docs/2.4/mod/core.htm l

Ogranicz dostęp przez IP

Wrażliwe obszary stron internetowych, takie jak panele kontrolne administratora, powinny mieć ścisłą kontrolę dostępu. Apache ułatwia umieszczanie na białej liście dostępu IP do niektórych lokalizacji Twojej witryny i odmawianie ruchu do wszystkich innych adresów IP. Możesz dodać następujące elementy do **pliku /etc/apache2/apache2.conf** .

```
<Katalog /ŚCIEŻKA/DO/WEBDIR/wp-admin >
    # zezwalaj na dostęp z jednego IP i dodatkowego zakresu IP,
    # i zablokuj wszystko inne
    Wymagaj ip 1.2.3.4
    Wymagaj ip 192.168.0.0/24
</Katalog>
```

W tym przykładzie użycie dyrektywy `Require` instruuje Apache, aby zezwolił na dostęp do określonej ścieżki, jeśli żądania pochodzą z któregokolwiek z wymienionych adresów IP, i odrzucił cały inny ruch.

Ogranicz dostęp za pomocą hasła

Dostęp do niektórych lokalizacji można również ustawić za pomocą poświadczeń opartych na haśle, korzystając z narzędzia `htpasswd` . Zobacz następujący artykuł, aby uzyskać więcej informacji:

- Hasło chroniące Twoją witrynę za pomocą pliku .htaccess

Zapobieganie atakom DoS

Domyślny model, w którym Apache przetwarza żądania (tzw. tryb prefork), podlega atakowi znanemu jako atak Slowloris . Atak Slowloris jest formą ataku DoS (Denial of Service), w którym serwer Apache jest zmuszony czekać na żądania od złośliwych klientów, co zajmuje dużo czasu, aby wysłać ruch, zmuszając w ten sposób uzasadnione żądania do przekroczenia limitu czasu lub całkowitego zignorowania. Na szczęście nowoczesne serwery Apache są w stanie złagodzić to zagrożenie za pomocą kilku dodatkowych dyrektyw konfiguracyjnych.

Włącz mod_reqtimeout

`mod_reqtimeout` to moduł Apache przeznaczony do zamykania połączeń od klientów, których wysłanie żądania trwa zbyt długo, na przykład w ataku Slowloris. Ten moduł zawiera dyrektywę, która pozwala Apache zamknąć połączenie, jeśli wyczuje, że klient nie wysyła danych wystarczająco szybko. Na przykład dodaj to do **pliku /etc/apache2/apache2.conf** :

```
Nagłówek RequestReadTimeout=10-20,MinRate=500 body=20,MinRate=500
```

W tym przykładzie Apache zamknie połączenie, jeśli klient potrzebuje więcej niż 10 sekund na wysłanie nagłówków HTTP lub jeśli klient potrzebuje więcej niż 20 sekund na wysłanie nagłówków z szybkością 500 bajtów na sekundę.

Apache zamknie również połączenie, jeśli klient potrzebuje więcej niż 20 sekund na wysłanie treści żądania, ale zezwoli na kontynuację żądania, o ile klient wysyła więcej niż 500 bajtów na sekundę.

Ta konfiguracja umożliwia klientom o niskiej jakości połączenia TCP (takim jak zdalni klienci z dużymi opóźnieniami lub w sieciach komórkowych lub satelitarnych niskiej jakości) wysyłanie żądań, jednocześnie chroniąc przed znanymi odciskami palców ataku Slowloris. `Konfiguracje RequestReadTimeout` mogą być złożone, dlatego zalecamy zapoznanie się z dodatkowymi informacjami na temat tej dyrektywy na stronie dokumentacji modułu .

Zabezpieczanie SSL/TLS

Upewnij się, że wszystkie witryny na Twoim serwerze mają zainstalowany certyfikat SSL. Możesz zainstalować bezpłatny certyfikat Let's Encrypt, postępując zgodnie z poniższymi instrukcjami:

- Konfigurowanie Let's Encrypt na DreamCompute z Apache

Wymuś wszystkie połączenia przez TLS

Jeśli masz zainstalowany certyfikat Let's Encrypt, jak pokazano powyżej, masz już możliwość wymuszenia całego ruchu przez SSL. Jeśli nie używasz certyfikatu Let's Encrypt, możesz również to zrobić, dodając następujące informacje do pliku konfiguracyjnego swojej witryny znajdującego się pod adresem **/etc/apache2/sites-available/example.com.conf** .

Najpierw upewnij się, że moduł `nagłówków` jest aktywny.

```
[użytkownik@instancja]$ sudo nagłówki a2enmod
[użytkownik@instancja]$ sudo restart usługi apache2
```

Następnie możesz dodać następujące elementy do pliku **.conf swojej witryny.**

```
<IfModule mod_headers.c>
Nagłówek zawsze ustawia Strict-Transport-Security "max-age=15768000; in
</IfModule>
```

Dla wszystkich połączeń w postaci zwykłego tekstu skonfiguruj Apache tak, aby wysyłał przekierowanie 301 dla żądań do wersji TLS witryny:

```
<Host wirtualny 192.168.1.1:80>
    [...]
    Nazwa serwera przyklad.com
```

```
    Przekierowanie stałe / https://example.com/
</Host wirtualny>
```

Dodatkowe środki bezpieczeństwa

Oprócz podstaw instalowania bezpiecznego pliku binarnego Apache, blokowania dostępu do wrażliwych obszarów witryny i prawidłowej obsługi połączeń TLS, istnieje kilka dodatkowych kroków, które może podjąć użytkownik szczególnie świadomy bezpieczeństwa:

Zainstaluj plik WAF

WAF (zapora sieciowa aplikacji) to oprogramowanie przeznaczone do sprawdzania ruchu HTTP/HTTPS, odrzucania złośliwych żądań i ogólnie działania jako dodatkowa warstwa bezpieczeństwa w stosie WWW HTTP. Odpowiednio skonfigurowany WAF może chronić Twoją witrynę przed atakami SQLi, XSS, CSRF i DDoS, a także zapewniać łagodzenie ataków siłowych i łatanie zagrożeń typu zero-day. Najpopularniejszym i najbardziej stabilnym WAF dla Apache jest ModSecurity . Wyświetl stronę GitHub projektu, aby uzyskać więcej informacji na temat instalacji i konfiguracji.

Automatyczna analiza logów + monitorowanie

Programy takie jak Fail2Ban mogą być używane do monitorowania dostępu do Apache i dzienników błędów, wyszukiwania wzorców ataków i podejmowania działań przeciwko atakującemu klientowi (takich jak zrzucanie adresów IP, zgłaszanie złośliwego zachowania właścicielowi adresu IP itp.). Fail2Ban jest rozszerzalny, co pozwala na tworzenie unikalnych wzorców wyszukiwania i zachowań odpowiedzi. Aby uzyskać więcej informacji i szczegółów dotyczących instalacji i konfiguracji, zobacz stronę GitHub projektu:

- Strona projektu fail2ban GitHub

Ogranicz ruch wejściowy za pośrednictwem IPTables

Oprócz zabezpieczenia samego Apache ważne jest zabezpieczenie środowiska hosta używanego do hostowania serwera WWW. Zablokowanie dostępu do rzeczy takich jak SSH może znacznie zwiększyć bezpieczeństwo hosta, zapobiegając próbom włamań. Powszechnym podejściem jest umieszczanie na białej liście znanych adresów IP, które będą uzyskiwać dostęp do hosta przez SSH i odrzucanie całego innego ruchu na porcie 22, lub użycie pola przeskoku, które ściśle filtruje dostęp do powłoki. Możesz to również zrobić, konfigurując niestandardową grupę zabezpieczeń dla swojej instancji. Zobacz następujący artykuł, aby uzyskać więcej informacji: