راهنمای ساخت گواهی خود اعتبار داده شده (self-signed) برای Apache در Ubuntu 20.04

مقدمه:

TLS یا "transport layer security" – و پیشین آن SSL – پروتکل­های برای پوشش ترافیک نرمال در یک رپر (wrapper) محافظت و کدگذاری شده است. با استفاده از این تکنولوژی سرورها قادر هستند که بدون آن که اطلاعات مورد رهگیری یا بازخوانی توسط سوم شخص قرار گیرد، آن­ها در امنیت را به متقاضیان خود ارسال کنند.

در این راهنما به چگونگی ساخت و استفاده از یک گواهی SSL خود اعتبار داده شده همراه با سرور وب Apache روی Ubuntu 20.04 می­پردازیم.

توجه داشته باشید: یک گواهی خود اعتبار داده شده تمامی مکالمات بین سرور شما و هر متقاضی دیگر را به کد تبدیل می­کند. با این وجود به این دلیل که این اعتبار از جانب هیچ مرجع اعتباردهی معتبری که به همراه مرورگرهای وب یا سیستم عامل معرفی شده­اند، داده نشده است و کاربران قادر به استفاده از این گواهی برای تائید خودکار سرور شما نیستند. در نتیجه کاربران شما هنگام بازدید از سایت­تان با یک خطا روبرو خواهند شد.

به دلیل این محدودیت این گواهی­ها برای محیط­های تولیدی و در خدمت عموم مناسب نیستند. کاربرد عمده این گواهی آزمایش کردن و یا امنیت­سازی برای یک سرویس غیرحیاتی و تک کاربره و یا گروه کوچکی از کاربران است که می­توانند اعتبار این گواهی را از طریق کانال­های ارتباطی جایگزین، تائید کنند.

برای یک راه حل معتبرتر می­توانید سری به مرجه رایگان "Let's Encrypt" بزنید.

پیش­نیازها:

برای ادامه این راهنما شما به مورد زیر نیاز دارید:

برای پیش­برد این راهنما یک سرور Ubuntu 20.04 نیاز دارید که طبق راهنمای تنظیمات اولیه سرور برای Ubuntu 20.04 تنظیم شده باشد. هم­چنین سرور شما می­بایست یک کاربر غیر root به همراه امتیازات sudo و یک فایروال پایه داشته باشد.

همچنین نیاز به Apache نصب شده دارید. می­توانید با استفاده از apt  Apache را دانلود کنید. شاخص پکیج محلی خود را به­روزرسانی کنید تا از تغییرات بالادستی مطلع شوید:

sudo apt update

سپس پکیج apache2  را نصب کنید:

* + sudo apt install apache2

در نهایت اگر فایروال ufw  را نصب دارید، پورت­های http  و https  را باز کنید:

* + sudo ufw allow "Apache Full"

بعد از انجام این مراحل، از ورود با کاربر غیر root خود اطمینان حاصل کنید و به راهنما را ادامه دهید.

گام 1- فعال­سازی **mod\_ssl**

پیش از آن که ما بتوانیم از هر گواهی SSL استفاده کنیم، باید mod\_ssl را که مدول Apache است و از کدگذاری SSL پشتیبانی می­کند، فعال کنیم.

با فرمان a2enmod  mod\_ssl را فعال کنید:

* sudo a2enmod ssl

برای فعال کردن مدول، Apache را ری­استارت کنید:

* sudo systemctl restart apache2

mod\_ssl اکنون فعال شده و آماده استفاده است.

گام2- ساخت گواهی SSL:

اکنون که Apache آماده استفاده است، می­توان به سخات یک گواهی SSL جدید پرداخت. این گواهی اطلاعاتی پایه درباره سایت شما ذخیره خواهد کرد. هم­چنین یک فایل کد در همراهی این گواهی خواهد بود که سرور را برای مدیریت امن اطلاعات کدگذاری شده، توان­مند می­سازد.

با فرمان openssl  می­توان کد SSL ساخت و فایل­ها را اعتبار بخشید:

sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/apache-selfsigned.key -out /etc/ssl/certs/apache-selfsigned.crt

بعد از وارد کردن فرمان شما به یک صفحه (prompt) هدایت خواهید شدکه می­توانید اطلاعاتی درباره وبسایت­تان وارد کنید. پیش از آن، به اجزای فرمانی که از آن استفاده می­کنیم می­پردازیم:

openssl: ابزار خط فرمانی است که برای ساخت و مدیریت گواهی­های OpenSSL، کدها و سایر فایل­ها استفاده می­شود.

req -x509: تمایل ما به استفاده از X.509 برای مدیریت درخواست صدور گواهی (CSR) را مشخص می­کند. X.509 یک زیرساخت کد عمومی استاندارد است که SSL و TLS برای مدیریت گواهی و کدها از آن بهره می­برند.

-nodes: این بخش به OpenSSL خواهد گفت که از گزینه ایمن­سازی گواهی ما با یک عبارت عبور، گذر کند. ما نیاز داریم که Apache بتواند فایل­ها را در شروع فعالیت سرور بدون هیچ دخالتی بخواند. از آن جایی که در صورت غیرفعال نکردن این گزینه نیاز به ورود هرباره عبارت گذر در شروع فعالیت سرور خواهد بود، از فعال­سازی آن چشم می­پوشیم.

-days 365: این مدت زمان اعتبار گواهی را بیان می­کند. این جا ما مدت زمان را یک اسل انتخاب کرده­ایم. بسیاری از مرورگرهای امروزی گواهی­های با اعتبار بیش از یک سال را قبول نمی­کنند.

-newkey rsa:2048: این قسمت بیان­گر آن است که مایل به ساخت هم­زمان یک گواهی و یک کد هستیم. از آن جایی که در مرحله قبل برای گواهی کدی نساختیم، نیاز به ساختن آن در موازات ساخت گواهی داریم. قسمت rsa:2048 مشخص می­کند که یک کد RSA با طول 2048 بایت ساخته شود.

-keyout: این خط به OpenSSL خواهد گفت که کجا فایل کد خصوصی که در حال ساخت آن هستیم، قرار دهد.

-out: این خط به OpenSSL خواهد گفت که کجا گواهی که در حال ساخت آن هستیم، قرار دهد.

پرامپت را بلافاصله پر کنید. مهم­ترین قسمت بخش Common Name است. شما باید یا اسم هاستی که برای ورود به سرور استفاده می­کنید و یا IP عمومی سرور را وارد کنید. نکته مهم این است که این اسم با آدرسی که در محل نشانه مرورگر خود برای ورود به سایت استفاده می­کنید هم­خوانی داشته باشد؛ چرا که عدم انطباق این دو خطای امنیتی را رقم خواهد زد.

لیست کامل پرامپت­ها چیزی شبیه به این خواهد بود:

Country Name (2 letter code) [XX]:US

State or Province Name (full name) []:Example

Locality Name (eg, city) [Default City]:Example

Organization Name (eg, company) [Default Company Ltd]:Example Inc

Organizational Unit Name (eg, section) []:Example Dept

Common Name (eg, your name or your server's hostname) []:your\_domain\_or\_ip

Email Address []:webmaster@example.com

هر دو فایلی که شما ساختید در زیرمجموعه لیست­های /etc/ssl قرار خواهند گرفت. در گام بعدی به به­روزرسانی تنظیمات Apache خواهیم پرداخت تا بتوانیم از گواهی و کد استفاده کنیم.

گام3- تنظیم Apache برای استفاده از SSL:

اکنون با در دسترس بودن گواهی و کد جدید، باید Apache را برای استفاده از آن­ها به­روزرسانی کنیم. در Ubuntu می­توانید فایل‌های تنظیمات Apache را ( که به .conf ختم می­شوند) در /etc/apache2/sites-available/قرار دهید و این تنظیمات بار دیگر که Apache بارگذاری یا مجددا آغاز می­شود، بارگذاری می­شوند.

برای این راهنما ما یک فایل تنظیماتی کوچک می­سازیم. (اگر از قبل تنظیمات <Virtualhost>  Apache را دارید و تنها نیاز به اضافه کردن SSL دارید، فایل­هایی که با SSL شروع می­شوند را کپی کنید و پورت VirtualHost  را از 80 به 443تغییر دهید. در گام بعدی به پورت 80 خواهیم پرداخت).

فایل جدیدی را در لیست /etc/apache2/sites-available directory باز کنید:

* sudo nano /etc/apache2/sites-available/your\_domain\_or\_ip.conf

تنظیمات زیر را الصاق کنید:

/etc/apache2/sites-available/your\_domain\_or\_ip.conf

<VirtualHost \*:443>

 ServerName your\_domain\_or\_ip

 DocumentRoot /var/www/your\_domain\_or\_ip

 SSLEngine on

 SSLCertificateFile /etc/ssl/certs/apache-selfsigned.crt

 SSLCertificateKeyFile /etc/ssl/private/apache-selfsigned.key

</VirtualHost>

از تغییر ServerName به هر آدرسی که مایل هستید سرور خود را با آن گزارش کنید، اطمینان حاصل کنید. این می­توانید اسم هاست، اسم کامل دامنه یا آدرس IP باشد. دقت کنید که انتخاب شما با Common Name  ، که حین ساخت گواهی استفاده کردید، مطابقت داشته باشد.

خطوط بعدی به تبیین آدرس منبع لیست SSL DocumentRoot  فایل­ها و گزینه­هایی خواهد پرداخت که که گواهی و کد جدید ما را به Apache نشان می­دهد.

اکنون به ساخت DocumentRoot  و قرار دادن فایل HTML در آن صرفا به منظور تست خواهیم پرداخت:

sudo mkdir /var/www/your\_domain\_or\_ip

 با ویرایشگر متن خود فایل index.html  را باز کنید:

sudo nano /var/www/your\_domain\_or\_ip/index.html

متن زیر را در فایل خالی الصاق کنید:

/var/www/your\_domain\_or\_ip/index.html

<h1>it worked!</h1>

واضحا این یک فایل HTML نیست؛ اما مرورگرها در این مورد سخت نمی­گیرند و این برای تایید تنظیمات شما کفایت می­کند.

فایل را ذخیره کنید و ببندید.

در مرحله بعد نیاز داریم تا فایل تنظیمات را با ابزار a2ensite  فعال کنیم:

sudo a2ensite your\_domain\_or\_ip.conf

در فرمان زیر به آزمایش خطاهای تنظیماتی میپردازیم:

sudo apache2ctl configtest

اگر همه چیز با موفقیت انجام شده باشد، نتیجه­ای مشابه زیر دریافت می­کنید:

Output

AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1. Set the 'ServerName' directive globally to suppress this message

خط اول پیامی است که به شما می­گوید دستور ServerName به صورت سراسری تنظیم نشده است. اگر می­خواهید از این پیام رها شوید، اسم ServerName خود را باسم دامنه سرور خود یا آدرس IP موجود در فایل /etc/apache2/apache2.conf تنظیم کنید. این مرحله دل بخواه است.

اگر خروجی شما شامل Syntax OK  می­شود، تنظیمات شما هیچ ایراد علمی ندارد. در امنیت می­توانیم Apache را مجددا بارگذاری کنیم تا تغییرات را اجرایی کنیم:

sudo systemctl reload apache2

اکنون سایت خود را درون مرورگر بالا بیاورید و برای این مورد از استفاده https://  در ابتدای آدرس اطمینان حاصل کنید.

باید یک خطا ببینید. برای یک گواهی خود اعتبار داده شده این امر، طبیعی است! مرورگر با این خطا به شما هشدار می­دهد که نمی­تواند هویت سرور را شناسایی کند؛ چرا که گواهی ما تحت تایید هیچ یک از مراجع مرورگر نیست. وجود این مسئله برای امور آزمایشی و شخصی بلامانع است. گزینه Advanced و سپس more information را انتخاب کنید تا ادامه دهید.

بعد از این مرحله مرورگز شما پیام موفقیت­آمیز it worked! را نشان خواهد داد.

توجه داشته باشد! اگر مرورگر به سرور متصل نشد، از نبود ممانعت یک فایروال اطمینان حاصل کنید. اگر از ufw استفاده می­کنید، فرمان زیر پورت­های 80 و 443را باز می­کند:

sudo ufw allow "Apache Full"

گام4- ریدایرکت دادن HTTP به HTTPS:

تا به اکنون تنظیمات ما تنها به درخواست­های HTTPS موجود بر پورت 443را پاسخ می­دهد. چنان­چه مایل به کد کردن تمامی ترافیک هستید، مفید خواهد بود اگر به درخواست­های موجود بر پورت 80هم پاسخ داده شود. به ساخت یک VirtualHost  برای پاسخ­دهی به این درخواست­های کدنشده و ریدایرکت­های HTTPS می­پردازیم.

فایل تنظیماتی Apache را که در گام قبلی باز کردیم، مجددا باز کنید:

sudo nano /etc/apache2/sites-available/your\_domain\_or\_ip.conf

در قسمت پایین، بخش VirtualHost  دیگری که با درخواست­های پورت 80 مطابقت می­کند، بسازید. مجددا از ServerName  برای انطباق اسم دامنه یا آدرس IP استفاده کنید. سپس از Redirect  برای مطابقت با هر درخواستی استفاده کنید و آن را به VirtualHost  SSL بفرستید. از نوشتن اسلشی که در دنباله آمده است، اطمینان حاصل کنید:

/etc/apache2/sites-available/your\_domain\_or\_ip.conf

<VirtualHost \*:80>

 ServerName your\_domain\_or\_ip

 Redirect / https://your\_domain\_or\_ip/

</VirtualHost>

بعد از اتمام، ذخیره کنید و فایل را ببندید. مجددا صحت و درستی تنظیمات خود را چک کنید و Apache را مجددا بالا آورید:

sudo apachectl configtest

sudo systemctl reload apache2

برای چک کردن صحت عملکرد ریدایرکت خود می­توانید از سایت خود با وارد کردن http:// در مقابل آدرس، دیدن کنید. شما می­بایست به­صورت خودکار به https://  ریدایرکت شوید.