راه‌اندازی و پیکربندی امن پروتکل SSL/TLS بر روی سرویس‌دهنده وب Nginx 1.10

شماره مستند APA-AMIRKABIR-13950710-1

تاریخ نگارش ..... ۱۰ مهر ۱۳۹۵

شماره نگارش ۱/۰

نگارش ..... آپای امیرکبیر

طبقه‌بندی ..... عادی

فهرست مطالب

۱ مقدمه

برای تأمین محرمانگی و جامعیت داده‌های مبادله شده می‌توان از پروتکل‌های استانداردی که بدین منظور طراحی شده استفاده کرد. در حال حاضر مهم‌ترین پروتکل رمزنگاری که در سطح اینترنت برای رمزنگاری داده‌های لایه کاربرد و تأمین امنیت ارتباطات استفاده می‌شود، پروتکل SSL/TLS است. در این گزارش مراحل نصب گواهی‌نامه SSL و امن‌سازی پروتکل SSL/TLS بر روی سرویس‌دهنده وب Nginx نسخه ۱,۱۰ بیان شده است.

۲ فعال‌سازی ارتباطات HTTPS

برای پیکربندی سرویس‌دهنده HTTPS و استفاده از این پروتکل ابتدا باید گواهی‌نامه دیجیتال مربوطه را از مراکز صدور گواهی (CA)[۱] معتبر دریافت کرد (یا گواهی خود-امضا را تولید کرد). گرفتن گواهی دارای مراحلی است که برای اطلاعات بیشتر در این زمینه می‌توانید به گزارش ارائه شده توسط پژوهشکده آپای دانشگاه صنعتی امیرکبیر که در آدرس زیر قرار دارد مراجعه کنید:

http://apa.aut.ac.ir/?p=971

در ادامه قصد داریم تا فایل پیکربندی Nginx را که در Ubuntu/Debian و RHEL/CentOS به ترتیب در مسیرهای زیر قرار دارد را به منظور استفاده از ارتباطات امن HTTPS و پیکربندی امن آن، ویرایش کنیم:

- `/etc/nginx/sited-enabled/yoursite.com`
- `/etc/nginx/conf.d/nginx.conf`

برای تمام موارد این گزارش، نیاز دارید که قسمت server از فایل مربوطه را ویرایش کنید.

توجه: قبل از انجام تغییرات روی این فایل، یک نسخه پشتیبان از آن تهیه کنید.

برای استفاده از ارتباطات HTTPS باید پارامتر ssl در قسمت server از فایل مربوط به پیکربندی فعال شود و همچنین مکان گواهی‌نامه سرویس‌دهنده و کلید خصوصی هم باید به صورت زیر در این فایل بیان شود.

```
server {
    listen              443 ssl;
    server_name         www.example.com;
    ssl_certificate     www.example.com.crt;
    ssl_certificate_key www.example.com.key;
    ssl_protocols       TLSv1 TLSv1.1 TLSv1.2;
    ssl_ciphers         HIGH:!aNULL:!MD5;
    ...
}
```

فایل گواهی (www.example.com.crt)، یک موجودیت عمومی (غیر محرمانه) است ولی فایل کلید (www.example.com.key) به صورت محرمانه نگهداری می‌شود. توجه کنید که امکان دارد گواهی و کلید خصوصی هر دو در یک فایل باشند که به صورت زیر بیان می‌شود و با این وجود باز هم فقط گواهی به سرویس‌گیرنده ارسال خواهد شد.

ssl_certificate_key www.example.com.cert;

زمانی که از یک مراکز صدور گواهی میانی، گواهی دریافت کنید، آن CA، زنجیره گواهی‌های میانی خود را در قالب یک بسته[۲] در اختیار شما قرار می‌دهد که باید با گواهی امضا شده سرویس‌دهنده ترکیب شود و در سرویس‌دهنده وارد شود. برای ترکیب آنها از دستور زیر استفاده می‌کنیم:

```
$ cat www.example.com.crt bundle.crt > www.example.com.chained.crt
```

و نتیجه را به صورت زیر در سرویس‌دهنده مورد استفاده قرار می‌دهیم:

[۱] Certificate Authority
[۲] Bundle

مرکز پژوهشی آپا (آگاهی‌رسانی، پشتیبانی، امداد برای آسیب‌پذیری‌ها و حوادث امنیتی سایبری)
تهران - بالاتر از چهارراه ولیعصر - نبش کوچه بالاور – ساختمان معاونت پژوهشی دانشگاه صنعتی امیرکبیر - طبقه سوم
کد پستی: ۱۵۹۱۶۳۴۳۱۱ تلفکس: ۶۶۴۶۰۳۰۸ Web: https://apa.aut.ac.ir - Email: autcert@aut.ac.ir

```
server {
    listen              443 ssl;
    server_name         www.example.com;
    ssl_certificate     www.example.com.chained.crt;
    ssl_certificate_key www.example.com.key;
    ...
}
```

اگر اشتباهی در این مورد وجود داشته باشد، Nginx خطای زیر را نشان خواهد داد:

```
SSL_CTX_use_PrivateKey_file(" ... /www.example.com.key") failed
   (SSL: error:0B080074:x509 certificate routines:
    X509_check_private_key:key values mismatch)
```

اگر می‌خواهید یک سرویس‌دهنده را طوری پشتیبانی کنید که هر دو درخواست HTTP و HTTPS را مدیریت کند باید مطابق زیر آن را پیکربندی کنید:

```
server {
    listen              80;
    listen              443 ssl;
    server_name         www.example.com;
    ssl_certificate     www.example.com.crt;
    ssl_certificate_key www.example.com.key;
    ...
}
```

توجه: برای اعمال تغییرات بالا و همچنین آنچه در ادامه گفته خواهد شد، باید بعد از تغییرات مورد نظر، به صورت زیر سرویس‌دهنده Nginx راه‌اندازی مجدد شود:

```
# Check the config first:
/etc/init.d/nginx configtest
# Then restart:
/etc/init.d/nginx restart
```

۳ پیکربندی امن پروتکل SSL/TLS

در این بخش چگونگی پیکربندی امن SSL/TLS را در سرویس‌دهنده وب Nginx بیان می‌کنیم. مواردی همچون استثنا کردن برخی الگوریتم‌های رمز به منظور کاهش حملاتی شبیه به FREAK، CRIME و LogJAM، غیرفعال سازی نسخه‌های ناامن SSL، برقرار کردن رمزنگاری‌های قوی که از Forward Secrecy (FS) پشتیبانی می‌کنند و فعال سازی HSTS را بیان می‌کنیم.

برای بررسی وضعیت امنیتی پروتکل SSL/TLS سرویس‌دهنده خود، می‌توانید به ابزاری که بدین منظور توسط پژوهشکده آپای دانشگاه صنعتی امیرکبیر طراحی شده و در آدرس زیر قرار دارد، مراجعه کنید.

https://sslcheck.certcc.ir

۳–۱ غیرفعال سازی SSLv2 و SSLv3

SSLv2 و SSLv3 (به خاطر حمله POODLE) ناامن هستند و باید غیرفعال شوند. برای غیرفعال سازی آنها، فایل مخصوص پیکربندی را به صورت زیر ویرایش می‌کنیم:

```
ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
```

۳–۲ غیرفعال سازی الگوریتم‌های رمزنگاری ضعیف

Forward Secrecy اطمینان می‌دهد که صحت[۱] یک کلید جلسه[۲] حتی وقتی که کلیدهای زیادی مورد مخاطره قرار گرفتند، حفظ می‌شود. FS کامل[۳] این مورد را با استخراج یک کلید جدید برای هر جلسه، به انجام می‌رساند. این بدان معناست که زمانی که کلید خصوصی به مخاطره افتاد، نمی‌تواند برای رمزگشایی ترافیک SSL مورد استفاده قرار گیرد.

پیشنهاد می‌شود دستور زیر را برای استفاده از الگوریتم‌های رمزنگاری قوی و غیرفعال سازی رمزنگاری‌های ضعیف در فایل پیکربندی وارد کنید. اگر نسخه OpenSSL شما قدیمی باشد، الگوریتم‌های غیرقابل دسترس به صورت خودکار دور انداخته می‌شوند. دقت کنید که همیشه از کل الگوریتم‌ها استفاده کنید و اجازه دهید تا OpenSSL، آنهایی را که پشتیبانی می‌کند انتخاب کند.

```
ssl_ciphers "EECDH+AESGCM:EDH+AESGCM:ECDHE-RSA-AES128-GCM-
SHA256:AES256+EECDH:DHE-RSA-AES128-GCM-SHA256:AES256+EDH:ECDHE-RSA-
AES256-GCM-SHA384:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-
SHA384:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA:ECDHE-RSA-AES128-
SHA:DHE-RSA-AES256-SHA256:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA:DHE-
RSA-AES128-SHA:ECDHE-RSA-DES-CBC3-SHA:EDH-RSA-DES-CBC3-SHA:AES256-GCM-
SHA384:AES128-GCM-SHA256:AES256-SHA256:AES128-SHA256:AES256-SHA:AES128-
SHA:DES-CBC3-SHA:HIGH:!aNULL:!eNULL:!EXPORT:!DES:!MD5:!PSK:!RC4";
```

[۱] Integrity
[۲] Session Key
[۳] Perfect Forward Secrecy

ترتیب الگوریتم‌ها خیلی مهم است زیرا الگوریتم‌ها به ترتیب انتخاب می‌شوند. لیست نوشته شده در بالا الگوریتم‌هایی که PFS را فراهم می‌آورند را در اولویت قرار داده است. نسخه‌های قدیمی‌تر OpenSSL ممکن است بعضی از الگوریتم‌های بالا را شامل نشود.

۳–۳ امن سازی پارامترهای دیفی هلمن

ما نیاز داریم تا یک پارامتر دیفی هلمن قوی را تولید کنیم، که می‌توانیم با دستور زیر این کار را انجام دهیم:

```
cd /etc/ssl/certs
openssl dhparam -out dhparam.pem 4096
```

و سپس باید به Nginx بگوییم که از این پارامترها برای تغییر کلید دیفی هلمن [۱] استفاده کند:

```
ssl_dhparam /etc/ssl/certs/dhparam.pem;
```

۴–۳ جلوگیری از OCSP Stapling

در زمان ارتباط با یک سرویس‌دهنده، سرویس‌گیرنده‌ها باید اعتبار گواهی سرویس‌دهنده را با استفاده از لیست خاصی به نام CRL [۲] یا پروتکل OCSP [۳] بررسی کنند. مشکل استفاده از CRL این است که این لیست بزرگ می‌شود و دریافت آن مشکل است. OCSP یک پروتکل است که برای بدست آوردن وضعیت یک گواهی دیجیتال X.509 استفاده می‌شود و در RFC 6960 بیان شده است.

برای امن‌سازی در مقابل این حمله باید تنظیمات زیر در قسمت server از فایل پیکربندی قرار داده شود.

```
ssl_stapling on;
ssl_stapling_verify on;
resolver 8.8.8.8 8.8.4.4 valid=300s;
resolver_timeout 5s;
```

۱–۴–۳ بررسی کارکرد

ابتدا یک ترمینال باز کنید و دستور OpenSSL زیر را برای اتصال به وب‌سایت خود وارد کنید.

```
openssl s_client -connect example.org:443 -tls1 -tlsextdebug -status
```

پاسخ را به صورت زیر مشاهده خواهید کرد:

```
OCSP response:
======================================
OCSP Response Data:
    OCSP Response Status: successful (0x0)
    Response Type: Basic OCSP Response
    Version: 1 (0x0)
    Responder Id: 99E4405F6B145E3E05D9DDD36354FC62B8F700AC
    Produced At: Feb  3 04:25:39 2014 GMT
    Responses:
    Certificate ID:
      Hash Algorithm: sha1
      Issuer Name Hash: 0226EE2F5FA2810834DACC3380E680ACE827F604
```

[۱] DHE key-exchange
[۲] Certificate Revocation List
[۳] Online Certificate Status Protocol

مرکز پژوهشی آپا (آگاهی‌رسانی، پشتیبانی، امداد برای آسیب‌پذیری‌ها و حوادث امنیتی سایبری)
تهران - بالاتر از چهارراه ولیعصر - نبش کوچه بالاور – ساختمان معاونت پژوهشی دانشگاه صنعتی امیرکبیر - طبقه سوم
کد پستی: ۱۵۹۱۶۳۴۳۱۱ تلفکس: ۶۶۴۶۰۳۰۸ Web: https://apa.aut.ac.ir - Email: autcert@aut.ac.ir

```
        Issuer Key Hash: 99E4405F6B145E3E05D9DDD36354FC62B8F700AC
        Serial Number: C1A3D8D00D72FCE483CD84759E9EC0BC
    Cert Status: good
    This Update: Feb  3 04:25:39 2014 GMT
    Next Update: Feb  7 04:25:39 2014 GMT
```

این بدان معنی است که این مورد به درستی کار می‌کند. اگر شما پاسخی شبیه زیر دریافت کردید، یعنی کار نمی‌کند:

```
OCSP response: no response sent
```

۳–۵ اضافه کردن سرآیند HSTS

در صورت امکان شما باید ویژگی HSTS[۱] را فعال کنید برای اینکه مرورگرها فقط با پروتکل HTTPS بتوانند با سایت شما ارتباط برقرار کنند.

برای فعال سازی HSTS باید دستور زیر را در قسمت server از فایل پیکربندی اضافه کنید:

```
add_header Strict-Transport-Security "max-age=63072000;
includeSubdomains; ";
```

۳–۶ جلوگیری از ClickJacking

به علت عدم ارسال سرآیندهای امنیتی مناسب توسط سرویس‌دهنده وب، این امکان برای یک مهاجم وجود دارد تا با طراحی یک صفحه وب که این سامانه به صورت یک IFrame در آن قرار داده شده، حمله‌ای به اسم ClickJacking را طراحی و اجرا نماید. در این حمله IFrame که در آن سایت هدف قرار داده شده با یک لایه فریبنده توسط مهاجم پنهان می‌شود و کاربر قربانی ترغیب می‌شود تا بر روی این لایه جدید کلیک نماید. کلیک بر روی این لایه در حقیقت بر روی Frameی که در زیر این لایه قرار دارد انجام می‌شود و کاربر ناخواسته اعمالی را بر روی سایت هدف (مثلاً سایت پرداخت اینترنتی) انجام می‌دهد. عدم وجود سرآیند X-Frame-Options در سرآیندهای پاسخ HTTP نشان‌دهنده وجود این آسیب‌پذیری است.

برای امن سازی سرویس‌دهنده Nginx در مقابل این حمله باید دستور زیر در قسمت server از فایل پیکربندی اضافه شود:

```
add_header X-Frame-Options "DENY";
```

[۱] HTTP Strict Transport Security

۴ منابع

1 http://nginx.org/en/docs/http/configuring_https_servers.html

2 https://www.digicert.com/ssl-certificate-installation-nginx.htm

3 https://raymii.org/s/tutorials/Strong_SSL_Security_On_nginx.html

4 https://wiki.mozilla.org/Security/Server_Side_TLS

5 http://nginx.org/en/docs/http/configuring_https_servers.html

6 https://juliansimioni.com/blog/https-on-nginx-from-zero-to-a-plus-part-2-configuration-ciphersuites-and-performance/

7 https://raymii.org/s/tutorials/HTTP_Strict_Transport_Security_for_Apache_NGINX_and_Lighttpd.html

مرکز پژوهشی آپا (آگاهی‌رسانی، پشتیبانی، امداد برای آسیب‌پذیری‌ها و حوادث امنیتی سایبری)
تهران - بالاتر از چهارراه ولیعصر - نبش کوچه بالاور – ساختمان معاونت پژوهشی دانشگاه صنعتی امیرکبیر - طبقه سوم
کد پستی: ۱۵۹۱۶۳۴۳۱۱ تلفکس: ۶۶۴۶۰۳۰۸ Web: https://apa.aut.ac.ir - Email: autcert@aut.ac.ir