راه‌اندازی و پیکربندی امن پروتکل SSL/TLS بر روی سرویس‌دهنده وب jetty 9

شماره مستند APA-AMIRKABIR-13950703-1

تاریخ نگارش ..... ۳ مهر ۱۳۹۵

شماره نگارش ۲/۰

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 را بر روی سرویس‌دهنده وب jetty نسخه 9.3.12 ارائه شده است.

۲ پیکربندی پروتکل SSL/TLS در Jetty

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

برای پیکربندی SSL در jetty، برخی از مراحل زیر باید انجام شود:

- تولید زوج کلید و گواهی‌نامه
- درخواست یک گواهی‌نامه قابل اعتماد
- بارگذاری کلیدها و گواهی‌نامه‌ها
- پیکربندی SslContextFactory در jetty

۲–۱ تولید زوج کلید و گواهی‌نامه

راه ساده برای تولید کلیدها و گواهی‌نامه‌ها استفاده از برنامه keytool است که گواهی‌نامه‌ها و کلیدها را مستقیما داخل keystore وارد می‌کند.

۲–۲ درخواست یک گواهی‌نامه قابل اعتماد

برای داشتن گواهی‌نامه‌ای که در مرورگرها قابل اعتماد باشد، باید گواهی‌نامه را از مراکز صدور گواهی(CA) معتبر تهیه کرد. ابتدا باید درخواست صدور گواهی (CSR) را با دستور زیر (با keytool) تهیه کرده و آن را به CA مورد نظر برای تولید گواهی ارائه دهید.

```
$ keytool -certreq -alias jetty -keystore keystore -file jetty.csr
```

۲–۳ بارگذاری کلیدها و گواهی‌نامه‌ها

بعد از اینکه گواهی‌نامه را از CA مورد نظر دریافت کردید یا اینکه خودتان آن را تولید کردید، باید آن را در JSSE keystore بارگذاری کنید.

۲–۳–۱ بارگذاری گواهی‌نامه‌ها با keytool

شما می‌توانید با استفاده از keytool، یک گواهی‌نامه را در keystore بارگذاری کنید. دستور زیر یک گواهی‌نامه با کدگذاری PEM را در فایل jetty.crt درون JSSE keystore بارگذاری می‌کند.

```
$ keytool -keystore keystore -import -alias jetty -file jetty.crt -trustcacerts
```

[۱] Certificate Authority

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

اگر گواهی‌نامه‌ای که از CA گرفته‌اید در فرمت قابل قبول keytool نباشد، می‌توانید با دستور زیر، فرمت آن را تغییر دهید.

```
$ openssl x509 -in jetty.der -inform DER -outform PEM -out jetty.crt
```

۲–۳–۲ بارگذاری گواهی‌نامه‌ها از طریق PKCS12

اگر شما یک کلید و گواهی در فایل جداگانه دارید، شما نیاز دارید تا آنها را در یک فایل با فرمت PKCS12 ترکیب کنید تا بتوانید در یک keystore جدید آن را بارگذاری کنید. توجه کنید که گواهی‌نامه می‌تواند توسط خود شما تولید شده باشد یا اینکه در پاسخ به یک درخواست CSR شما، از طرف CA برای شما صادر شده باشد.

دستور OpenSSL زیر، دو فایل کلید و گواهی را با نام‌های jetty.key و jetty.crt می‌گیرد و آنها را داخل یک فایل به نام jetty.pkcs12 قرار می‌دهد.

```
$ openssl pkcs12 -inkey jetty.key -in jetty.crt -export -out jetty.pkcs12
```

اگر شما زنجیره‌ای از گواهی‌ها دارید (وقتی که CA شما یک CA میانی است)، باید فایل PKCS12 را به صورت زیر بسازید:

```
$ cat example.crt intermediate.crt [intermediate2.crt] ... rootCA.crt > cert-chain.txt
$ openssl pkcs12 -export -inkey example.key -in cert-chain.txt -out example.pkcs12
```

توجه کنید که ترتیب گواهی‌ها باید از سرور تا ریشه باشد.

بعد از مراحل بالا، باید یک پسورد (غیر تهی) را برای OpenSSL وارد کنید و سپس فایل نهایی PKCS12 را با keytool در JSSE keystore بارگذاری کنید.

```
$ keytool -importkeystore -srckeystore jetty.pkcs12 -srcstoretype
PKCS12 -destkeystore keystore
```

توجه: اگر شما در حال به‌روزرسانی پیکربندی به منظور استفاده از گواهی‌نامه جدید هستید(در زمانی که گواهی‌نامه قبلی انقضا شده است)، فقط باید گواهی جدید را به آن صورت که در بخش بارگذاری گواهی‌نامه بیان شده است، بارگذاری کنید. اگر شما کلید و گواهی‌نامه را با استفاده از روش PKCS12 وارد کردید، از یک اسم مستعار "۱" به همراه "jetty" استفاده کنید.

۲–۴ پیکربندی SslContextFactory

گواهی‌های SSL تولید شده در بالا که در keystore قرار دارند، به عنوان یک نمونه از شی SslContextFactory پیکربندی شده‌اند.

در کل SslContextFactory مسئول:

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

- ساخت SslEngine جاوا که توسط اتصال‌دهنده‌های Jetty و سرویس‌گیرنده‌های Jetty استفاده می‌شود.
- مدیریت دسترسی Keystore
- مدیریت دسترسی Truststore
- مدیریت انتخاب پروتکل از طریق لیست Excludes / Includes
- مدیریت انتخاب رمزنگاری از طریق لیست Excludes / Includes
- پشتیبانی OCSP[1]
- و ...

ماژول‌های ارائه شده برای https و http2 به صورت خودکار SslContextFactory را مناسب SslConnectionFactory نصب می‌کنند و ServerConnectors را در جهت درست هدایت می‌کنند.

یک مثال از این نصب:

```
$ cd /path/to/mybase
$ java -jar /path/to/jetty-dist/start.jar --add-to-start=https
INFO: ssl              initialised (transitively) in
${jetty.base}/start.ini
INFO: https            initialised in ${jetty.base}/start.ini
INFO: Base directory was modified
$ ls -l
drwxrwxr-x.  2 user group  4096 Feb  2 11:47 etc/
-rw-rw-r--.  1 user group  4259 Feb  2 11:47 start.ini
$ ls -l etc
-rw-rw-r--.  1 user group  3697 Feb  2 11:47 keystore
```

زمانی که شما start.ini را بررسی کنید، مشاهده خواهید کرد که تنظیمات بیان شده زیادی برای شما در رابطه با پیکربندی اصول اولیه SslContextFactory در نظر گرفته شده است.

که برخی از آنها را در زیر بیان می‌کنیم:

jetty.ssl.host

پیکربندی اینکه اتصال دهنده‌های SSL/TLS باید روی چه رابطی[2] گوش فرا دهند.

jetty.ssl.port

پیکربندی اینکه اتصال دهنده‌های SSL/TLS باید روی چه پورتی گوش فرا دهند.

jetty.sslContext.keyStorePath

مکان keystore که شما آن را با گواهی خود پیکربندی کرده‌اید را تنظیم می‌کند.

jetty.sslContext.keyStorePassword

رمز عبور را برای keystore تنظیم می‌کند.

[1] Online Certificate Status Protocol

[2] Interfaces

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

۳ امن‌سازی پروتکل SSL/TLS در Jetty

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

در این قسمت چگونگی فعال و غیرفعال سازی استفاده از الگوریتم‌های رمزنگاری را بیان می‌کنیم. استفاده از برخی الگوریتم‌ها و پروتکل‌های رمزنگاری دارای آسیب‌پذیری‌هایی می‌باشند که باید غیرفعال شوند. به عنوان مثال برای جلوگیری از حمله BEAST لازم است تا مجموعه‌ای خاص از الگوریتم‌های رمزنگاری را پیکربندی کنیم.

هر دو دستور setIncludeCipherSuites و setExcludeCipherSuites می‌توانند نام دقیق الگوریتم‌های رمزنگاری در JDK و همچنین عبارات منظم برای اسم را مورد استفاده قرار دهند. اگر نیاز دارید تا این موارد را تنظیم کنید، بهتر است این موارد در یک فایل XML برای پیکربندی SslContextFactory انجام شود.

توجه: زمانی که با دستورات Includes / Excludes کار می‌کنید، همیشه Excludes اولویت دارد.

برای انجام این مورد، ابتدا یک فایل جدید (${jetty.base}/etc/tweak-ssl.xml) بسازید (می‌تواند هر نامی داشته باشد، فقط پیشوند "jetty-" نداشته باشد.)

```
<!DOCTYPE Configure PUBLIC "-//Jetty//Configure//EN"
          "http://www.eclipse.org/jetty/configure_9_3.dtd">
<!-- Tweak SsslContextFactory Includes / Excludes -->
<Configure id="sslContextFactory"
class="org.eclipse.jetty.util.ssl.SslContextFactory">
  <!-- Mitigate SLOTH Attack -->
  <Call name="addExcludeCipherSuites">
    <Arg>
      <Array type="String">
        <Item>.*_RSA_.*SHA1$</Item>
        <Item>.*_RSA_.*SHA$</Item>
        <Item>.*_RSA_.*MD5$</Item>
      </Array>
    </Arg>
  </Call>
</Configure>
```


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

این فایل XML جدید، شناسه sslContextFactory را پیکربندی خواهد کرد (آن شناسه ابتدا توسط ماژول ssl و ${jetty.home}/etc/jetty-ssl-context.xml مرتبط با آن ساخته شده است).

برای اینکه اطمینان حاصل کنید که ${jetty.base} شما از این فایل جدید XML استفاده می‌کند، می‌توانید آن را به آخر هر کدام از فایل‌های زیر اضافه کنید:

- ${jetty.base}/start.ini
- ${jetty.base}/start.d/server.ini

```
$ cd /path/to/mybase
$ ls -l
drwxrwxr-x.  2 user group  4096 Feb  2 11:47 etc/
-rw-rw-r--.  1 user group  4259 Feb  2 11:47 start.ini
$ tail start.ini
# Module: https
--module=https
etc/tweak-ssl.xml
$
```

۳-۲ تنظیمات امنیتی دیگر

در این بخش، تنظیمات امنیتی دیگری که با دستورات Include / Exclude قابل انجام است را بیان می‌کنیم.

مثال ۱: شامل شدن تمام الگوریتم‌های رمزنگاری که از Forward Secrecy پشتیبانی می‌کنند (با استفاده از عبارات منظم):

```
<!-- Enable Forward Secrecy Ciphers.
       Note: this replaces the default Include Cipher list -->
  <Set name="IncludeCipherSuites">
    <Array type="String">
      <Item>TLS_DHE_RSA.*</Item>
      <Item>TLS_ECDHE.*</Item>
    </Array>
  </Set>
```

مثال ۲: استثنا کردن تمام رمزنگاری‌های ناامن:

```
<!-- Eliminate Old / Insecure / Anonymous Ciphers -->
  <Call name="addExcludeCipherSuites">
    <Arg>
      <Array type="String">
        <Item>.*NULL.*</Item>
        <Item>.*RC4.*</Item>
        <Item>.*MD5.*</Item>
        <Item>.*DES.*</Item>
        <Item>.*DSS.*</Item>
      </Array>
    </Arg>
  </Call>
```

مثال ۳: از سال ۲۰۱۴ به بعد، SSLv3 ناامن شناخته شده است و باید غیرفعال شود.

```
<!-- Eliminate Insecure Protocols -->
```

```
<Call name="addExcludeProtocols">
  <Arg>
   <Array type="java.lang.String">
     <Item>SSL</Item>
     <Item>SSLv2</Item>
     <Item>SSLv2Hello</Item>
     <Item>SSLv3</Item>
   </Array>
  </Arg>
</Call>
```

توجه: غیرفعال کردن SSLv3 از اتصال مرورگرهای قدیمی نظیر IE6 روی ویندوز XP، جلوگیری می‌کند.

مثال ۴: از مذاکرات دوباره[۱] TLS می‌تواند جلوگیری شود تا از حمله مبتنی بر آن جلوگیری شود.

```
<Set name="renegotiationAllowed">FALSE</Set>
```

توجه: بعد از ذخیره کردن تغییرات در دستورات بالا، باید سرویس مورد نظر را راه‌اندازی مجدد کنید.

[۱] Renegotiation

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

۴ منابع

1 https://wiki.eclipse.org/Jetty/Howto/CipherSuites

2 https://support.sonatype.com/hc/en-us/articles/213465798-How-to-Configure-HTTPS-Cipher-Suites-Used-By-Nexus

3 https://www.eclipse.org/jetty/documentation/9.3.x/configuring-ssl.html

4 https://issues.sonatype.org/browse/NEXUS-7595

5 https://www.eclipse.org/jetty/documentation/9.4.x/configuring-ssl.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