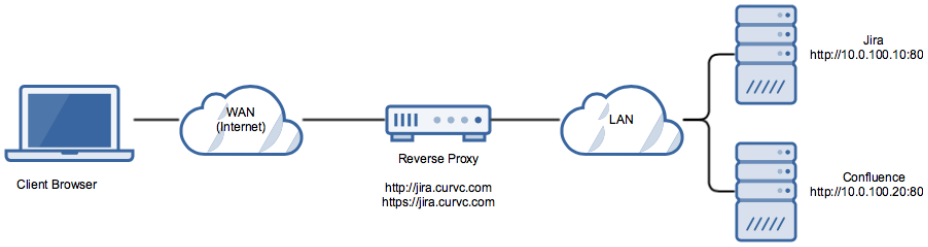
Atlassian 제품 HTTPS Proxy 구성하기

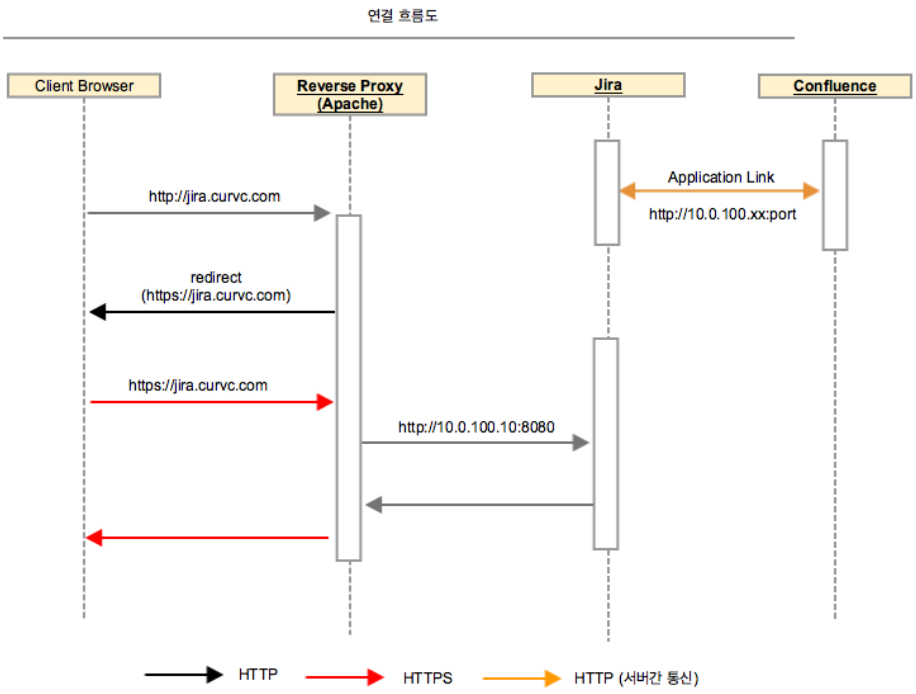
이 문서는 Apache HTTPD 서버를 이용해 하나 또는 여러개의 Atlassian 제품을 접속하는 통일된 HTTPS URL 주소를 구성하는 방법을 제공한다.

Overview

네트웍 구성도



접속 흐름 보기



사전 조건

도메인 및 SSL 인증서

- 도메인: Atlassian application을 위한 도메인 등록 (또는 sub-domain, 예: jira.curvc.com)
- 인증서: 복수개의 Atlassian 솔루션 사용시 sub-domain 에 공통으로 사용 할 수 있는 wildcard 도메인 구매 추천

Reverse Proxy 구성할 서버 준비

- OS: CentOS 7 (본 예제에서 사용)
- Apache HTTP 서버 설치 (2.4 이상)
    - mod_ssl 설치

Application context 사용하지 않음

본 예에서는 http://curvc.com/jira 와 같이 "jira" context를 사용하지 않음을 가정한다. Context를 사용하여 reverse proxy 를 구성할 수 있지만 드문 구성이다.

CentOS SELinux policy 수정

- SELinux가 httpd (80, 443) 프로세스의 network 접속을 금지하기 때문에 허용하도록 설정

```
# > sudo /usr/sbin/setsebool -P httpd_can_network_connect 1
```

Step 1) 인증서 준비

필요한 인증서 관련 파일을 준비한다. 인증 기관별로 제공되는 파일 설명을 참조하여 준비한다 (SSL 인증서 발급 및 적용).

- 키 파일: 인증서 발급할 때 사용했던 private key file
- 인증서: 인증기관에서 발급된 인증서
- chain file: intermediate 파일로부터 생성

Step 2) Reverse Proxy 구성

본 문서는 reverse proxy로 Apache httpd를 사용하여 설명한다.

```
    .
    .
    .
NameVirtualHost *:80
Listen 80
    .
    .
    .
IncludeOptional conf.d/*.conf
IncludeOptional sites-enabled/*.conf
```

```
    .
    .
    .
NameVirtualHost *:443
Listen 443 https
    .
    .
    .
<VirtualHost _default_:443>
   SSLEngine on
   SSLProtocol all -SSLv2 -SSLv3
   SSLCipherSuite HIGH:3DES:!aNULL:!MD5:!SEED:!IDEA
   SSLCertificateFile /etc/ssl/certs/STAR_curvc_com.crt
   SSLCertificateKeyFile /etc/ssl/certs/curvc.key
   SSLCertificateChainFile /etc/ssl/curvc_com.ca-bundle
</VirtualHost>
```

```
<VirtualHost *:80>
    ServerName jira.curvc.com

    RemoteIPHeader X-Forwarded-For
    ProxyPreserveHost   On
    RewriteEngine On
    ProxyVia Off

    <Proxy *>
         Order deny,allow
         Require all granted
    </Proxy>

    Redirect            "/" "https://jira.curvc.com/"
</VirtualHost>

<VirtualHost *:443>
    ServerName jira.curvc.com

    ProxyPreserveHost   On
    RewriteEngine On
    ProxyVia Off

    <Proxy *>
         Order deny,allow
         Require all granted
    </Proxy>

    ProxyPass           "/" "http://10.0.100.10:8080/"
    ProxyPassReverse    "/" "http://10.0.100.20:8080/"
    RemoteIPHeader X-Forwarded-For

    SSLEngine on
    SSLCertificateKeyFile /etc/ssl/certs/curvc.key
    SSLCertificateFile /etc/ssl/certs/STAR_curvc_com.crt
    SSLCertificateChainFile /etc/ssl/certs/curvc_com.ca-bundle

</VirtualHost>
```

```
<VirtualHost *:80>
    ServerName confluence.curvc.com

    ProxyPreserveHost   On
    RewriteEngine On
    ProxyVia Off

    <Proxy *>
         Order deny,allow
         Require all granted
    </Proxy>

    Redirect             "/" "https://confluence.curvc.com/"
    RemoteIPHeader X-Forwarded-For
</VirtualHost>

<VirtualHost *:443>
    ServerName confluence.curvc.com

    ProxyPreserveHost   On
    ProxyRequests Off
    RewriteEngine On

    SSLEngine on
    SSLProxyEngine on
    SSLCertificateKeyFile /etc/ssl/certs/curvc.key
    SSLCertificateFile /etc/ssl/certs/STAR_curvc_com.crt
    SSLCertificateChainFile /etc/ssl/certs/curvc_com.ca-bundle

    ProxyVia Off

    <Proxy *>
         Order deny,allow
         Require all granted
    </Proxy>

    RewriteEngine On
    RewriteCond %{REQUEST_URI} !^/synchrony
    ProxyPass           "/synchrony" "http://10.0.100.20:8091/synchrony"

    ProxyPass           "/" "http://10.0.100.20:8090/"
    ProxyPassReverse    "/" "http://10.0.100.20:8090/"

     <Location /synchrony>
        Require all granted
        RewriteEngine on
        RewriteCond %{HTTP:UPGRADE} ^WebSocket$ [NC]
        RewriteCond %{HTTP:CONNECTION} Upgrade$ [NC]
        RewriteRule .* ws://10.0.100.20:8091%{REQUEST_URI} [P]
    </Location>
</VirtualHost>
```

구성 완료 후 적용

```
$ > apache testconfig
$ > apache restart
```

Step 3) JVM에 certificate trustStore 지정

Application1 - (HTTPS)→ Application2 의 경우 Application 2 신뢰토록 설정

> 인증서를 시스템 공용으로 등록해도 된다면 Application이 사용하는 JVM keystore에 등록한다.
>
> 일반적인 keystore 파일 위치: <JVM install dir>/jre/lib/security/cacerts

Step 1)의 인증서와 key를 이용해 keyStore 생성 (wildcard 인증서라면 Atlassian application 공통 keystore 사용 권장)

keystore 생성 방법 참고 (이미 발급된 인증서를 이용해 SSL 구성하기)

...

CATALINA_OPTS="-Djavax.net.ssl.trustStore=/data/atlassian/application-data/confluence/atlassian.jks -Djavax.net.ssl.trustStorePassword=password ${CATALINA_OPTS}"

...

FISHEYE_OPTS="-Djavax.net.ssl.trustStore=/atlassian/application-data/atlassian.jks -Djavax.net.ssl.trustStorePassword=password ${FISHEYE_OPTS}"

```
bservmgr.exe //ES/AtlassianBitbucket

Java  >> Java Options:

-Djavax.net.ssl.trustStore=D:₩Atlassian₩ApplicationData₩atlassian.jks
-Djavax.net.ssl.trustStorePassword=password
```

Step 3) Tomcat Connector 수정

Atlassian application을 종료하고 tomcat의 server.xml 파일을 수정한다.

Bitbucket 5.0 이상을 제외한 Atlassian application의 경우:

HTTPS를 지원하지 않는 Application과 연결해야 한다면 이를 위한 Connector 추가 (예: 8180)

Connector directive

```
<!-- Additional connector for Application link -->
<Connector port="8180" connectionTimeout="20000" maxThreads="200" minSpareThreads="10"
enableLookups="false" acceptCount="10" URIEncoding="UTF-8" />

<Connector port=<default>
        maxThreads=<default>
    minSpareThreads=<default>
    connectionTimeout=<default>
    enableLookups=<default>
    maxHttpHeaderSize=<default>
    protocol=<default>
    useBodyEncodingForURI=<default>
    redirectPort=<default>
    acceptCount=<default>
    disableUploadTimeout=<default>
        proxyName="<subdomain>.<domain>.com"
        proxyPort="443"
        secure="true"
        scheme="https"/>
```

line 16 ~ 18 추가

- proxyName: 제공될 application URL (예: jira.curvc.com)

Bitbucket 5.0 이상의 경우:

링크를 참고하여 설정한다.

- 경로: <Bitbucket data home>/application-data/bitbucket/shared/bitbucket.properties

bitbucket.properties

```
server.secure=true
server.scheme=https
server.proxy-port=443
server.ssl.enabled=false
server.proxy-name=bitbucket.curvc.com
```

설정 완료 후 application을 기동한다.

Step 4) Atlassian Application Base URL 변경

Application 별로 base URL을 https://~ 로 변경한다.

예)

- Jira = https://jira.curvc.com
- Confluence = https://confluence.curvc.com

Step 5) Atlassian Application link 수정

HTTPS 지원 Application link

Jira와 Confluence Application link 예)

- Jira에서 Confluence URL 설정 = https://confluence.curvc.com
- Confluence에서 Jira URL 설정 = https://jira.curvc.com

HTTPS 미지원 Application link

연결하는 Application이 HTTPS 를 지원하지 않은 경우 server.xml에 구성한 Application link용 port를 사용하여 application link를 구성한다.

Jira와 Confluence Application link 예)

- Jira에서 Confluence URL 설정 = http://10.0.100.20:8190
- Confluence에서 Jira URL 설정 = http://10.0.100.10:8180

Apache HTTPD 2.2 고려사항 (CentOS 6)

2.2 환경은 2.4 다른 구성 방법을 적용한다

- Premission 지정 방법: "Allow from all" 사용
- `RemoteIPHeader X-Forwarded-For  :`

```
<VirtualHost *:80>
    ServerName jira.curvc.com

    RemoteIPHeader X-Forwarded-For
    ProxyPreserveHost   On
    RewriteEngine On
    ProxyVia Off

    <Proxy *>
         Order deny,allow
         Allow from all
    </Proxy>

    Redirect            "/" "https://jira.curvc.com/"
</VirtualHost>

<VirtualHost *:443>
    ServerName jira.curvc.com

    ProxyPreserveHost   On
    RewriteEngine On
    ProxyVia Off

    <Proxy *>
         Order deny,allow
         Allow from all
    </Proxy>

    ProxyPass           "/" "http://10.0.100.10:8080/"
    ProxyPassReverse    "/" "http://10.0.100.20:8080/"

    SSLEngine on
    SSLCertificateKeyFile /etc/ssl/certs/curvc.key
    SSLCertificateFile /etc/ssl/certs/STAR_curvc_com.crt
    SSLCertificateChainFile /etc/ssl/certs/curvc_com.ca-bundle

</VirtualHost>
```

Confluence Apache VirtualHost 예)

- Synchrony 고려

```
# Put this after the other LoadModule directives
LoadModule proxy_module /usr/lib/apache2/modules/mod_proxy.so
LoadModule proxy_http_module /usr/lib/apache2/modules/mod_proxy_http.so
LoadModule proxy_wstunnel_module /usr/lib/apache2/modules/mod_proxy_wstunnel.so
LoadModule rewrite_module /usr/lib/apache2/modules/mod_rewrite.so

# Put this in the main section of your configuration (or virtual host, if using Apache virtual hosts)

  ProxyRequests Off
  ProxyPreserveHost On

  RewriteEngine On
  RewriteCond %{REQUEST_URI} !^/synchrony
  RewriteRule ^/(.*) http://<domain>:8090/$1 [P]

  <Proxy *>
      Require all granted
  </Proxy>

  ProxyPass /synchrony http://<domain>:8091/synchrony

  <Location /synchrony>
      Require all granted
      RewriteEngine on
      RewriteCond %{HTTP:UPGRADE} ^WebSocket$ [NC]
      RewriteCond %{HTTP:CONNECTION} Upgrade$ [NC]
      RewriteRule .* ws://<domain>:8091%{REQUEST_URI} [P]
  </Location>

  ProxyPass / http://<domain>:8090
  ProxyPassReverse / http://<domain>:8090

  <Location />
      Require all granted
  </Location>
```