



Comodo SSL Manual

Index

01
인증 방법

02
SSL

03
아파치 설정

04
톰캣 설정

01 인증 방법

https://www.securesign.kr/
이 사이트가 정말 저렴하다. 아래 그림을 보자

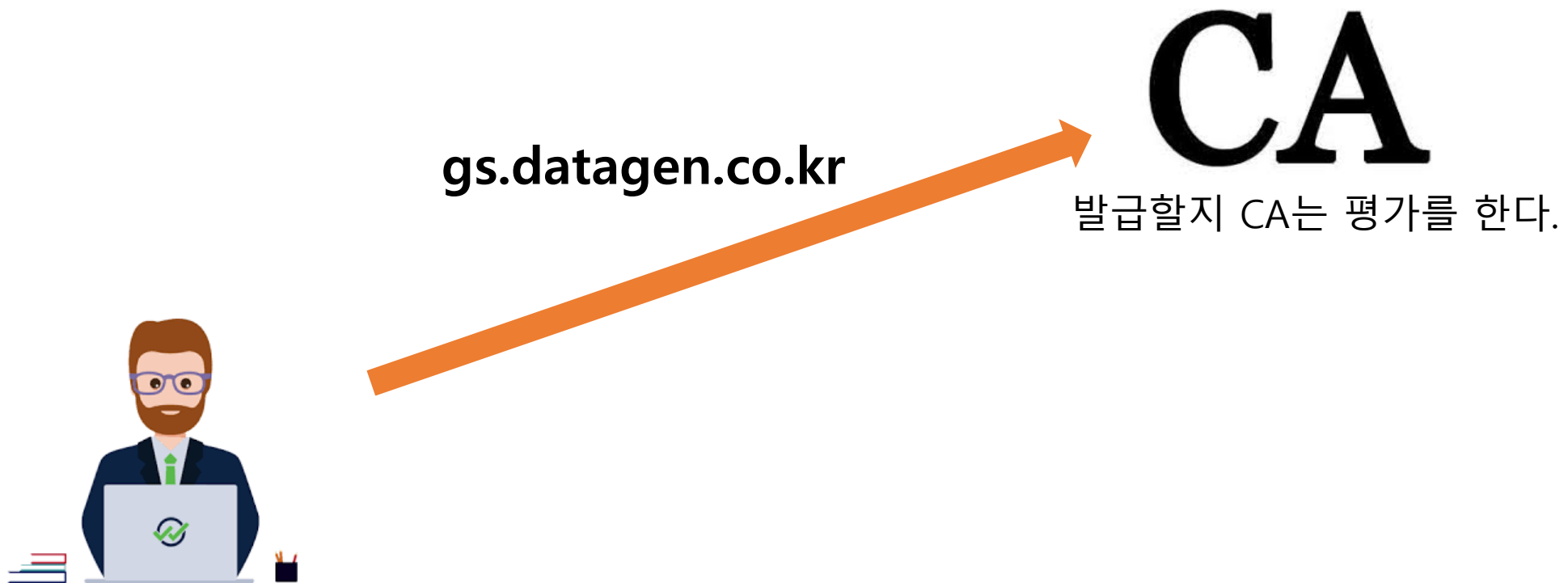
국내 최저가 수준! 가장 빠른 발급! 글로벌 산업 표준 SSL/TLS 보안서버 인증서!

🌐🌐🌐🌐🌐 주요 웹브라우저 99% 및 아이폰/안드로이드 호환 SSL 인증서

COMODO	RapidSSL.
PositiveSSL	Standard
₩ 1 만 2천원	₩ 2 만 6천원
도메인 1 개 배상금 \$ 10,000	도메인 1 개 배상금 \$ 10,000
자세히보기	자세히보기

소규모 웹사이트 및 스타트업 또는 네트워크 보호가 필요한 곳에 적합한 제품들이다.

인증서를 받기전 에 SSL인증서를 받고자 하는 도메인이 CA로부터 평가를 받아야 한다.
평가받는 인증방법을 DCV(Domain Control Vaildation)이라 한다.



그다음 페이지에서 평가를 받는법을 말하겠다.



각 CA마다 인증 받는 방식이 다르다는걸 알아두자.
우리가 검사를 맡아야 하는 CA는 COMODO 이다.
그외 CA는 구매하려는 사이트 매뉴얼을 잘 참조하자. ^^

DCV인증 방법중에 HTTP/s 인증방법을 소개하겠다.

우리의 웹서버는 아파치인 만큼 또 방법이 다르다.
다른 서버를 쓰고있다면 그것도 역시 매뉴얼을 참고해라!

HTTP/s 인증 방법

가상호스트(도메인)를 등록한 파일에 **DocumentRoot**가 있을것이다.
아래 그림을 보자. 아래 빨간 박스가 경로다.

```
<VirtualHost *:80>
    #DocumentRoot /home/datauser/www
    DocumentRoot /var/www/html
    ServerName gs.datagen.co.kr
    ErrorLog "/etc/httpd/logs/gs.datagen.co.kr-error.log"
    CustomLog "/etc/httpd/logs/gs.datagen.co.kr-access.log" common

    # ProxyRequests Off
    # PAC 를 리다이렉션 할 때 HOST 정보를 함께 전달 한다
```

가상호스트 파일 위치는 /etc/httpd/conf.d/vhost.conf 이다.

이제 저 경로(/var/www/html)로 들어가준다...

HTTP/s 인증 방법

들어간 경로에 디렉토리 2개를 생성해야한다. 즉
결론 은 아래와 같이 나와야 한다.

```
[root@gs pki-validation]# pwd  
/var/www/html/.well-known/pki-validation
```

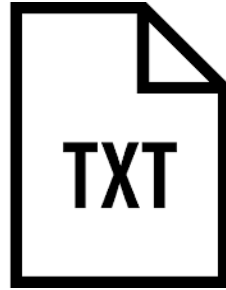


마지막 폴더(**pki-vaildation**)에 comodo를 제공하는 회사 (seuresign)에서 보낸 번호로 파일을 만들어야 한다.
말이 어렵다,.. 다음 페이지에서 보자.

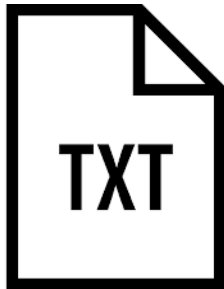

```
[root@gs pki-validation]# ls
19423DD5724462F8B367806800A9FFBC.txt
```



pki-vaildation



회사에서 준 코드.txt



회사에서 준 코드.txt



```
EC1A9208C408C47EB2937B08A3B763A11C0F7CE8146570D404120FA4F3A712D
comodoca.com
dcv201806259e97
~
~
~
~
```

그 파일 안에는 회사(securesign)에서 적으라고 하는 코드를 입력하면 된다.

오타나 띄어쓰기를 잘못한다면 **CA에서 잘못된 코드로 인식해 검증에 실패한다.**

CA는 안전한 도메인이라고 판단을 했다..



CA

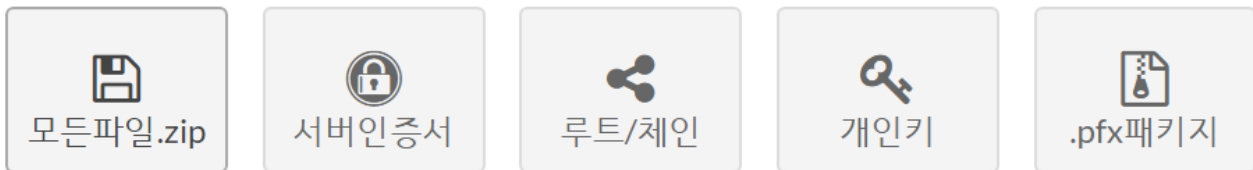


gs.datagen.co.kr

소요시간은 평균 30분이며, 도메인이름에 금융 관련 단어 다국적 기업명, 공공질서 위험 단어가 있는경우 2틀정도 심사를 받게 된다.

인증에 성공을 했으면 인증서를 발급 받을 수 있다.
아래 사진과 같은 인증서가 나오고 아파치에 적용을 해주면 끝이다.

📁 인증서 다운로드



서버 인증서 - gs_datagen_co_kr.crt (PEM)

개인키 - gs_datagen_co_kr.sha256rsa.key (PEM) / 암호없음

PFX,JKS (인증서 통합 패키지) - gs_datagen_co_kr.pfx / 암호 : 99e99p

체인 인증서 - COMODO RSA Domain Validation Secure Server CA (PEM)

* 체인 인증서 설치/적용은 필수 사항입니다

* .key / .pfx / .jks 파일은, CSR 자동 생성 신청시에만 제공됩니다.

* .pfx 및 .jks 파일에는 "개인키+서버인증서+체인인증서+루트인증서"가 포함되어 있습니다.

* 인증서 발급 내역서는 .zip 파일에 PDF 파일로 포함되어 있습니다.

여기까지가 회사에서 제공하는 서비스이고 그 이후 아파치 작업은 개발자에 몫이다.

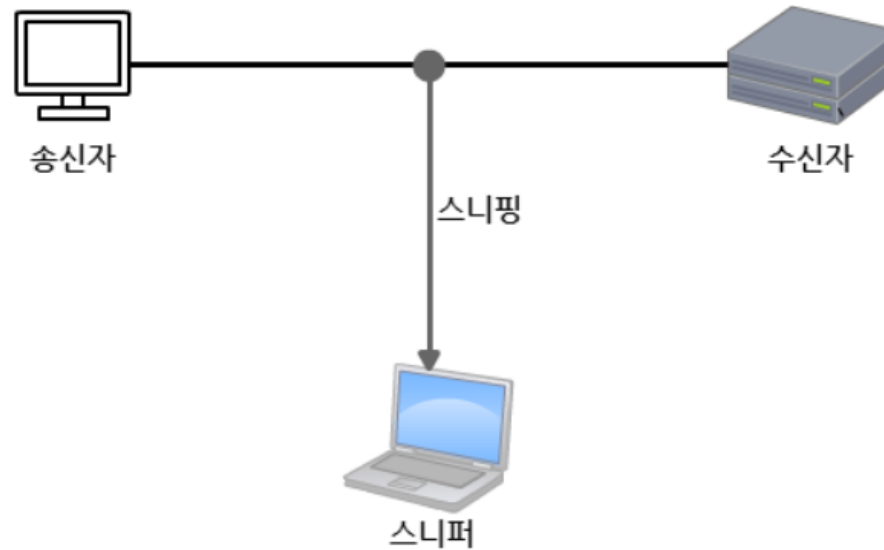


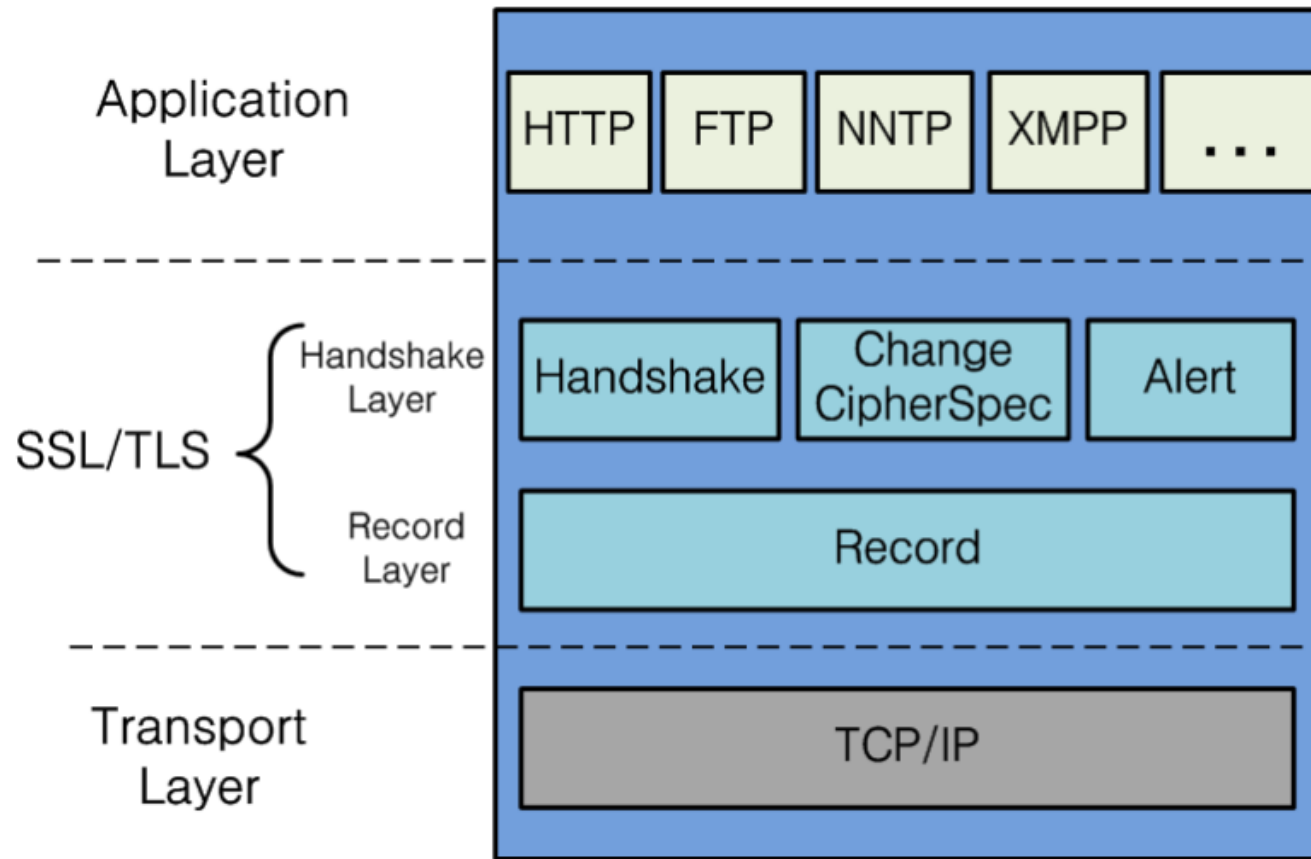
02
SSL

발급받은 파일은 무엇이며 SSL개념에 대해서 설명하겠다.

SSL 통신은 인터넷 상에서 **통신할 때 주고받는 데이터를 보호**하기 위한 표준화된 암호화 프로토콜

SSL을 사용하면 아래 그림과 같은 **스니핑**이라는 보안공격을 방어할 수 있다.





SSL은 전송계층이 다르기때문에 HTTP뿐만아니라 FTP, XMPP 등 응용계층 **프로토콜 종류에 상관없이 사용할 수 있다.**

SSL의 가장 주된 적용 대상이 HTTPS이다 보니 SSL과 HTTPS를 혼용하는 경우가 많다. 보안이라는 개념은 같으나, 위 사진에서 보여주듯이 둘은 다르다.

그럼 저 좋은 HTTPS가 적용된 SSL프로토콜은 아무나 사용할 수 있는가?

그건 아니다. 인증 방법에서 언급했듯이

CA(인증기관)에서 인증서를 받아야 통신을 할 수 있다.

즉, CA라는 기관에서 받은 **인증서가 있어야 SSL이 적용된 HTTPS를 사용할 수 있다.**

아래 그림은 대표적인 CA기관들이다.

순위	발행자	사용률	시장 점유율
1	코모도	16.7%	38.4%
2	IdenTrust	13.9%	32.0%
3	시만텍	5.6%	12.9%
4	고 대디	3.3%	7.5%
5	글로벌사인	1.9%	4.5%
6	DigiCert	1.0%	2.2%
7	Certum	0.3%	0.7%
8	Entrust	0.2%	0.4%
9	세콤	0.1%	0.3%
10	Actalis	0.1%	0.3%
11	Trustwave	0.1%	0.2%
12	Let's Encrypt	0.1%	0.2%
13	StartCom	0.1%	0.2%
14	WISeKey Group	< 0.1%	0.1%

인증받은 도메인은 CA로부터 인증서를 받는다.



gs.datagen.co.kr



CA
안전하군!

그럼 저 인증서에는 무슨내용이 있을까? 다음장에서 살펴 보겠다.

SSL 인증서에는 다음과 같은 정보가 포함되어 있다.

1. **서버의 정보** (인증서를 발급한 CA, 서비스의 도메인)
2. **서버측 공개키** (공개키 내용, 공개키의 암호화 방법)

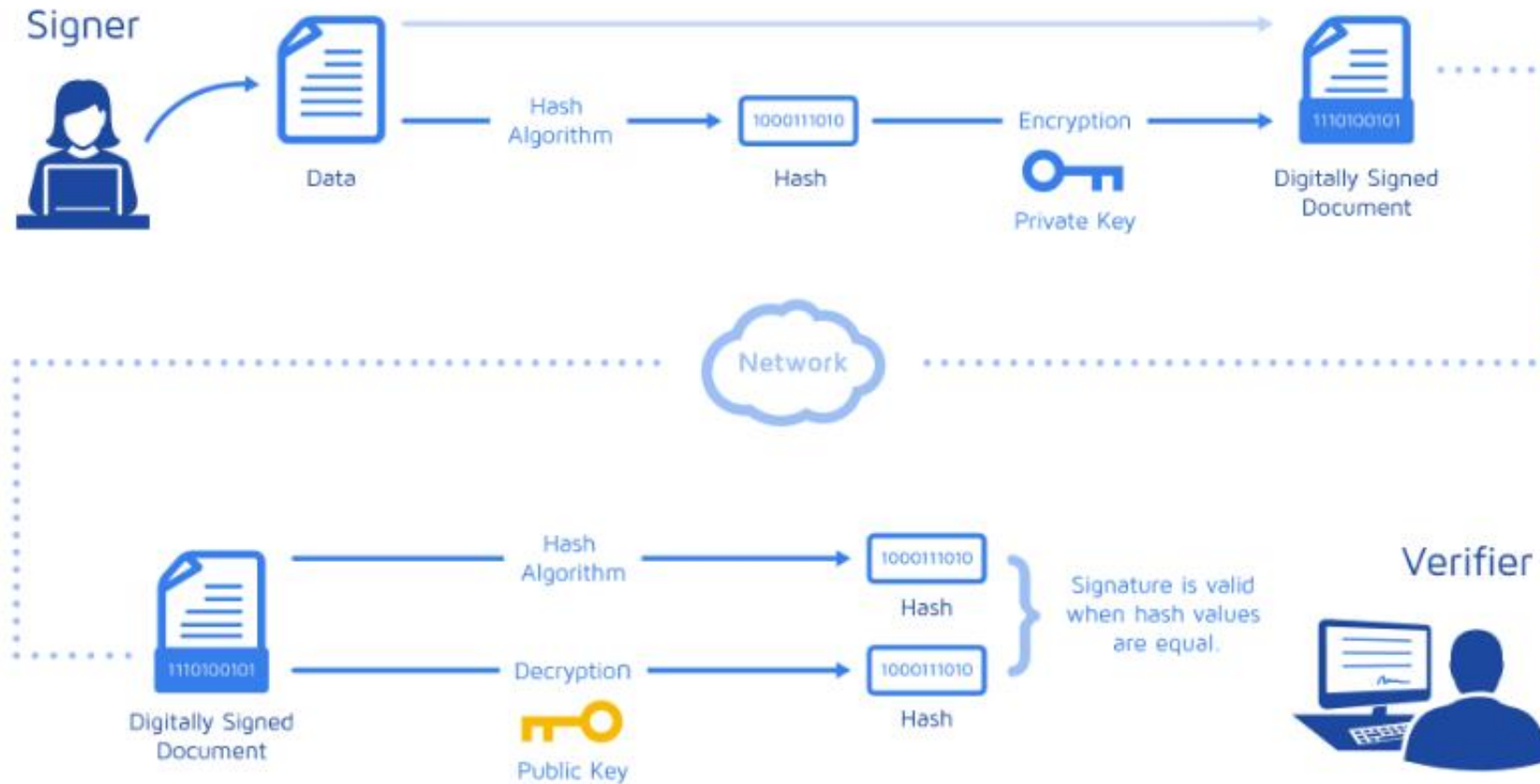
크게 2가지로 구분할 수 있다. 첫번째 서버의 정보는 클라이언트가 의도한 서버가 맞는지게 대한 내용이고 두번째는 서버와 통신할때 사용할 공개키와 그 공개키의 암호화 방법들의 정보를 담고 있다.

다음장에서는 인증서 동작 방법을 설명하겠다.



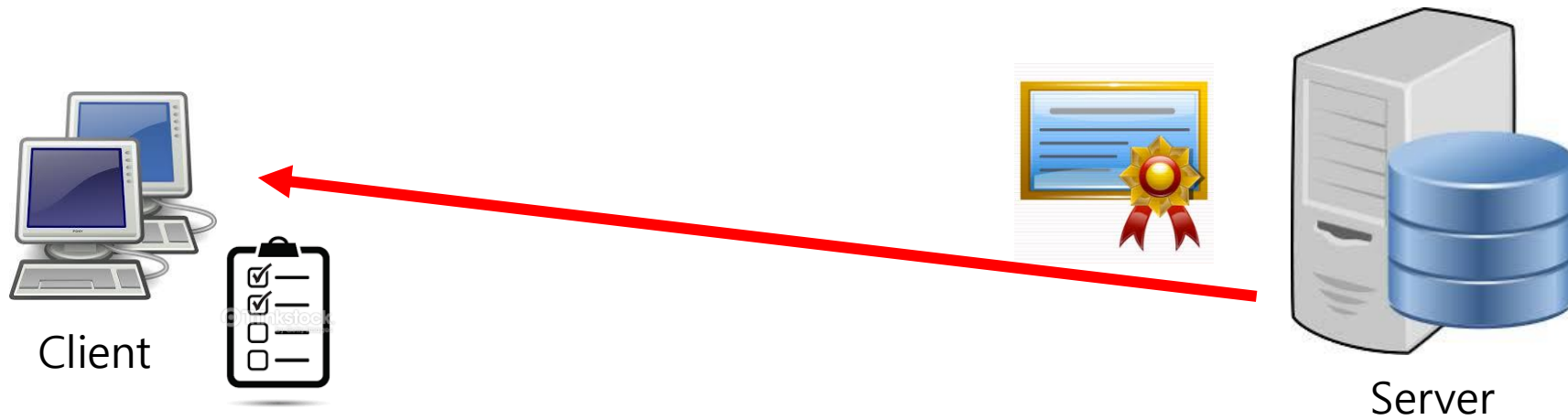
HTTPS /SSL 에서 사용하는 암호화는 대부분 **공개키방식**이다.
공개키 는 쉽게말해
A로 암호화 하면 B로 복호화할 수 있고
B로 암호화 하면 A로 복호화 하는 방식이다.

다음장 그림으로 좀더 이해해 보자.



Singer가 비밀키로 암호화를 하면 공개키와 함께 암호화된 정보를 Verifier전송한다.
정보와 공개키를 획득한 Verifier는 공개키를 이용해 복호화한다.
이 과정에서 공개키가 유출된다면 의도치 않은 공격자에 의해서 데이터가 복호화 될 위험이 있다.
하지만 데이터를 보호하는 것이 목적이 아니다. 공개키를 가지고 복호화 할 수 있다는 것은
그 데이터가 **공개키와 쌍을 이루는 비공개키에 의해서 암호화 되었다는 것을 의미**한다.
이러한 것을 **전자 서명**이라고 부른다.

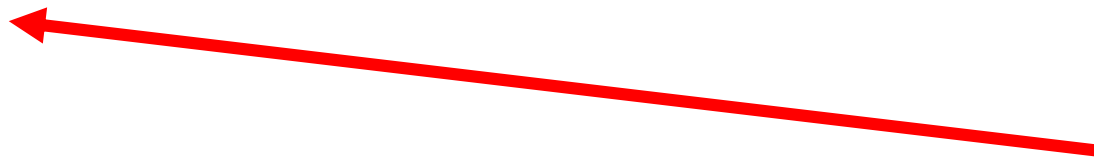
SSL이 사용하는 암호방식을 전 페이지에서 알게 되었다.
이제는 클라이언트가 서버에 접속했을때 과정을 보겠다.



Client는 서버의 인증서가 CA에 의해서 발급된 것인지를 확인하기 위해서 **클라이언트**
에 내장된 CA리스트를 확인한다.

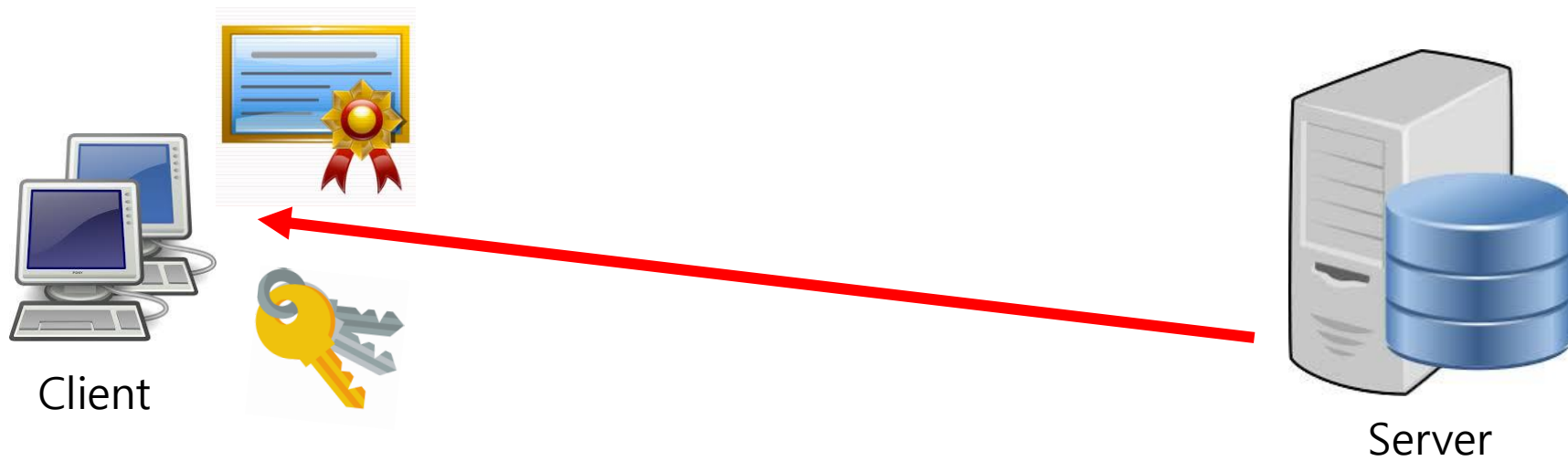


Client



Server

CA 리스트에 인증서가 없다면 사용자에게 경고 메시지를 출력한다.



리스트에 있다면 클라이언트에 내장된 CA의 공개키를 이용해서 인증서를 복호화 한다.



복호화에 성공했다면 인증서는 **CA의 개인키로 암호화된 문서임이 암시적으로 보증된 것**이다. 인증서를 전송한 서버를 믿을 수 있게 된다.

자, 이제 우리가 CA로 발급받은 파일들을 살펴보자

```
[root@gs datagenSSL]# ll
합계 72
drwx----- 2 root root 209 6월 26 20:26 RootChain
-rw-r--r-- 1 root root 2436 6월 26 20:14 gs.datagen.co.kr_201806259E97.crt.pem
-rw-r--r-- 1 root root 7181 6월 26 20:15 gs.datagen.co.kr_201806259E97.jks
-rw-r--r-- 1 root root 1702 6월 26 05:56 gs.datagen.co.kr_201806259E97.key.pem
-rw-r--r-- 1 root root 46223 6월 26 20:15 gs.datagen.co.kr_201806259E97.pdf
-rw-r--r-- 1 root root 7732 6월 26 20:14 gs.datagen.co.kr_201806259E97.pfx
[root@gs datagenSSL]#
```

crt.pem 은 인증서 정보이다. 즉, comodo 한테 인증을 받고 날라온 인증서이다. 저 인증서에는 크게 2가지가 있다고 전페이지에서 설명을 했다.

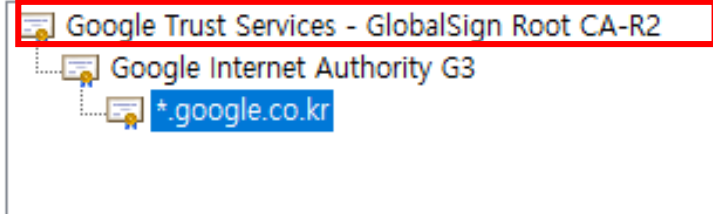


아래 key는 서버측 개인키다. 절대 노출이 되면 안된다!!
저걸로 암호화 한것을 사용자는 공개키로 복호화 하는것이다.
공개키로 복호화가 났다면 클라이언트는 서버를 신뢰하게 된다.

다음장에서는 Rootchain 폴더를 보겠다.


```
AddTrustExternalCARoot.crt  
COMODORSAddTrustCA.crt  
COMODORSADomainValidationSecureServerCA.crt  
ca-bundle.pem  
chain-bundle.pem
```

AddTrustExternalCARoot.crt는 RootCA의 인증서이다. 자체 서명으로 루트 인증서는 나야~ 라고 알려주는것이다. 즉 아래 사진처럼 가장위에 나온것이 rootCA이다.



마지막 chain-bundle.pem은 인증서 체인이라고 하며, 위 사진을 보면 google.co.kr 위에 있는 중간 CA(Google Internet Authority G3)이다.

간단한 HTTPS가 이루어지는 것을 살펴 보았다.
공학적으로 깊게 알고자 한다면 아래 사이트를 추천합니다..ㅎㅎㅎ

<http://www.moserware.com/2009/06/first-few-milliseconds-of-https.html>

하나 알게된 사실은 SSL -> TLS로 이름이 바졌다고 한다.

위키백과 참고

https://ko.wikipedia.org/wiki/%EC%A0%84%EC%86%A1_%EA%B3%84%EC%B8%B5_%EB%B3%B4%EC%95%88

03

아파치 설정

Open SSL



아파치에서 SSL통신을 하려면 **open_ssl**과 **mod_ssl**이 필요하다.
YUM으로 설치를 해주자. open_ssl은 기본으로 설치되어 있다.

mod_ssl을 설치하면 conf.d디렉토리 아래 **ssl.conf**가 생성이 된다.
앞으로 https로 통신할 가상호스트(DNS)를 저기에 등록해주고 설정해주면 된다.
또한 http에서 https로 리다이렉션 설정을 해줘야한다. 과정은 아래와 같다.



인증받은 파일을 서버로 옮긴다. 경로를 하나 생성해준다. 우리 경로는 아래와 같다.

```
[root@gs datagenSSL]# pwd  
/etc/comodo/ssl/datagenSSL
```

datagenSSL경로에 발급받은 인증서를 넣어준다.

```
[root@gs datagenSSL]# ll  
합 계 72  
drwx----- . 2 root root 209 6월 26 20:26 RootChain  
-rw-r--r-- . 1 root root 2436 6월 26 20:14 gs.datagen.co.kr_201806259E97.crt.pem  
-rw-r--r-- . 1 root root 7181 6월 26 20:15 gs.datagen.co.kr_201806259E97.jks  
-rw-r--r-- . 1 root root 1702 6월 26 05:56 gs.datagen.co.kr_201806259E97.key.pem  
-rw-r--r-- . 1 root root 46223 6월 26 20:15 gs.datagen.co.kr_201806259E97.pdf  
-rw-r--r-- . 1 root root 7732 6월 26 20:14 gs.datagen.co.kr_201806259E97.pfx  
[root@gs datagenSSL]#
```

RootChain -> CA리스트 파일(체인인증서, 루트인증서 가 여기 파일에 있음)

.cet.pem -> 디지털 인증서

.key.pem -> 서버쪽 비공개키

가상호스트 파일(vhost.conf)에 리다이렉션 프로퍼티를 넣어준다. 아래 박스를 보자.

```
<VirtualHost *:80>
    #DocumentRoot /home/datauser/www
    DocumentRoot /var/www/html
    ServerName gs.datagen.co.kr
    ErrorLog "/etc/httpd/logs/gs.datagen.co.kr-error.log"
    CustomLog "/etc/httpd/logs/gs.datagen.co.kr-access.log" common

    # on은 리다이렉션을 하겠다. off는 안 하겠다.
    RewriteEngine on
    RewriteCond %{SERVER_NAME} =gs.datagen.co.kr
    RewriteRule ^ https://%{SERVER_NAME}%{REQUEST_URI} [END,NE,R=permanent]
</VirtualHost>
```

RewriteEngine을 on으로 하면 Rewrite 모듈을 사용할 수 있다.
RewriteCond 규칙을 통해 해당 도메인 이름을 검사하고 RewriteRule에서 https로 리다이렉션 해준다.

http 요청시 자동으로 https로 변환된다. (단, 443 포트 방화벽이 열려 있다는 가정하에)

마지막으로 ssl.conf에서 발급받은 인증서를 입력해주면 끝이다.
설정해야 하는 목록은 다음페이지 에서 설명하겠다.

```

55
56 <VirtualHost *:443>
57
58 # General setup for the virtual host, inherited from global configuration
59 DocumentRoot "/var/www/html"
60 ServerName gs.datagen.co.kr
61

```

1. 443을 받을 DocumentRoot 경로와 서버이름을 다시 적어둔다. (vhost.conf 와 동일하게)

```

77 # SSL Cipher Suite:
78 # List the ciphers that the client is permitted to negotiate.
79 # See the mod_ssl documentation for a complete list.
80 SSLCipherSuite HIGH:3DES:!aNULL:!MD5:!SEED:!IDEA

```



```

77 # SSL Cipher Suite:
78 # List the ciphers that the client is permitted to negotiate.
79 # See the mod_ssl documentation for a complete list.
80 # SSLCipherSuite HIGH:3DES:!aNULL:!MD5:!SEED:!IDEA
81 SSLCipherSuite ECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH

```

2. (이부분은 필수사항은 아니다.) 사용할 알고리즘을 선택하는 부분이다.
SSL을 더욱더 보안강화 하기위해 아래와같은 알고리즘으로 수정한다.
수정하는이유는 MD5 알고리즘은 매우 취약하다고 한다.

받은 인증서를 넣어준다.

```
101 SSLCertificateFile /etc/comodo/ssl/datagenSSL/gs.datagen.co.kr_201806259E97.crt.pem
102
103 #   Server Private Key:
104 #   If the key is not combined with the certificate, use this
105 #   directive to point at the key file.  Keep in mind that if
106 #   you've both a RSA and a DSA private key you can configure
107 #   both in parallel (to also allow the use of DSA ciphers, etc.)
108 SSLCertificateKeyFile /etc/comodo/ssl/datagenSSL/gs.datagen.co.kr_201806259E97.key.pem
109
110 #   Server Certificate Chain:
111 #   Point SSLCertificateChainFile at a file containing the
112 #   concatenation of PEM encoded CA certificates which form the
113 #   certificate chain for the server certificate. Alternatively
114 #   the referenced file can be the same as SSLCertificateFile
115 #   when the CA certificates are directly appended to the server
116 #   certificate for convinience.
117 SSLCertificateChainFile /etc/comodo/ssl/datagenSSL/RootChain/chain-bundle.pem
118
119 #   Certificate Authority (CA):
120 #   Set the CA certificate verification path where to find CA
121 #   certificates for client authentication or alternatively one
122 #   huge file containing all of them (file must be PEM encoded)
123 SSLCACertificateFile /etc/comodo/ssl/datagenSSL/RootChain/AddTrustExternalCARoot.crt
```

서버인증서

개인키

체인인증서

루트인증서

남은 프로퍼티들은 상황에 맞게 설정하면된다. 기본설정은 끝이다.
우리는 프록시 기능을 추가해야한다. 아무 라인에 맞게 다음 페이지 처럼 넣어둔다.



```
132
133     #프록시 설정
134     ProxyRequests Off
135     # WAS 로 리다이렉션 할 때 HOST 정보를 함께 전달 한다 .
136     ProxyPreserveHost On
137
138     ProxyPass /care http://111.112.113.11:8080/care
139     ProxyPassReverse /care http://111.112.113.11:8080/care
140
141     ProxyPass /news http://111.112.113.12:8080/news
142     ProxyPassReverse /news http://111.112.113.12:8080/news
143
144     ProxyPass /api http://111.112.113.12:8080/api
145     ProxyPassReverse /api http://111.112.113.12:8080/api
146
147     ProxyPass /admin http://111.112.113.12:8080/admin
148     ProxyPassReverse /admin http://111.112.113.12:8080/admin
149
150     ProxyPass /shop http://111.112.113.13:8080/shop
151     ProxyPassReverse /shop http://111.112.113.13:8080/shop
152
153     ProxyPass /image http://111.112.113.15:8080/image
154     ProxyPassReverse /image http://111.112.113.15:8080/image
155
```

위와 같이 vhost.conf에서 설정한 프로퍼티들을 넣어둔다.
단 **https(433)로만 통신할때만**이다.

```
[root@gs ~]# apachectl configtest  
Syntax OK
```

혹시모를 오타를 위해 테스트를 하고 아파치를 restart해준다.



← → ↻  안전함 | <https://gs.datagen.co.kr/admin/action/main/loginForm>

아파치 설정끝 ㅎㅎ

04 톰캣 설정

아파치도 했으니 톰캣도 해보자.
구매방법과 인증방법은 아파치나 톰캣이나 같다. (생략)
대신 발급받은 인증서를 톰캣은 **server.xml**에 넣으면 된다는 것이다.

아래와 같은 경로를 만들어준다.

```
[root@dapchain dapchainSSL]# pwd  
/etc/comodo/ssl/dapchainSSL
```

경로에 발급받은 인증서를 아파치랑 똑같이 넣어준다.

Config 파일 아래 server.xml을 들어가자.
아래와 같은 태그를 찾아 아래와 같이 수정하자.

```
92     <Connector port="443" protocol="org.apache.coyote.http11.Http11NioProtocol"  
93               maxThreads="150" scheme="https" secure="true" SSLEnabled="true" keystoreFile="/etc/comodo/ssl/dap  
chainSSL/dapchain.net_20180626I764.jks" keystorePass="3ux1pz" clientAuth="false" sslProtocol="TLS">  
94     </Connector>  
95
```

톰캣은 jsk(인증서 통합 패키지)를 넣어주고 발급받은 비밀번호 securesign에 있음.
이미 server.xml 리다이렉션 기능이 **디폴트로 설정**되어 있다.

web.xml 만 수정하면 리다이렉션 기능이 완벽하게 구동된다.

```
<security-constraint>
  <web-resource-collection>
    <web-resource-name>Restricted URLs</web-resource-name>
    <url-pattern>/*</url-pattern>
  </web-resource-collection>
  <user-data-constraint>
    <transport-guarantee>CONFIDENTIAL</transport-guarantee>
  </user-data-constraint>
</security-constraint>
```

web-app 태그 안에 넣어준다. 그리고 톰캣을 재시작하면 끝

각 태그에 설명은 아래와 같다.

<security-constraint> 태그는

"URL 매핑을 사용하여 자원 컬렉션에 대한 액세스 특권을 정의하는 데 사용됩니다." 이라 api에 써있다. 액세스 특권을 받아야 리다이렉션이 되는것 같다.

<web-resource-collection> 태그는 "보호 할 리소스 세트를 설명하는 URL 패턴 (제한하려는 호스트 이름 및 포트 다음의 URL 부분) 및 HTTP 조작 (제한하려는 URL 패턴과 일치하는 파일 내의 메소드) . 웹 리소스 컬렉션은 웹 리소스 컬렉션 지정에서 설명."

<user-data-constrain> "클라이언트와 서버간에 전송 될 때 데이터가 보호되는 방법을 지정합니다. 사용자 데이터 제약 조건은 보안 연결 지정에 설명되어 있습니다."

참고 : <https://docs.oracle.com/cd/E19798-01/821-1841/bncbk/index.html>

