

오픈소스 라이브러리 배포 이야기

(feat. Maven Central
Repository)

표준프레임워크 오픈커뮤니티
딸기아빠

Contents

- Open Source Library 소개
- Maven Central Repository 개요
- OSS Repository Hosting 활용
- CI(Continuous Integration) 적용



Open Source Library 소개

Code Analyst

'17년 5월부터 개발,
'18년 10월 및 11월 오픈소스 버전 릴리즈

Code Analyst Project

release v2.8.0 build passing license Apache-2.0

<https://github.com/RedCA-Family/code-analyst>

sonarqube

eclipse

IntelliJ IDEA

범례

오픈소스
공개

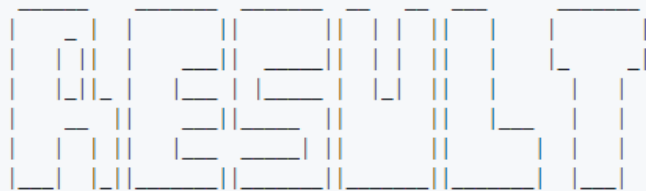
오픈소스
준비중

- 다양한 코드품질 관련 지표를 통합적으로 측정 및 제공 (CLI 및 API)
 - 코드 규모, 중복도, 복잡도, 잠재적 결함, 응집도, 결합도, 개발표준 등
- 현재 Java, JavaScript(Node.js), C# 및 Python 언어 지원

Code Analyst – examples

```
14:28:28.929 INFO c.s.a.c.m.App - Project Directory : C:\Workspace\Project
14:28:28.935 INFO c.s.a.c.m.App - Code Size Analysis start...
```

...



Files : 60
Dir. : 13
Classes : 66
Functions : 540
lines : 6,703
Comment Lines : 384
Ncloc : 4,633
Statements : 2,210

Duplicated Blocks : 40
Duplicated lines : 50
Duplication % : 0.75%

Complexity functions : 524
Complexity Total : 932
Complexity Over 10(%) : 0.76% (4)
Complexity Over 15(%) : 0.38% (2)
Complexity Over 20(%) : 0.00% (0)
Complexity Equal Or Over 50(%) : 0.00% (0)
- The complexity is calculated by PMD's Modified Cyclomatic Complexity method

SonarJava violations : 447
SonarJava 1 priority : 3
SonarJava 2 priority : 43
SonarJava 3 priority : 339
SonarJava 4 priority : 62
SonarJava 5 priority : 0

RedCA Dashboard

Size ?

Lines Of Code

1,440

Lines
3,067

Files

47

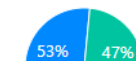
Directories
9

Functions

159

Classes
43

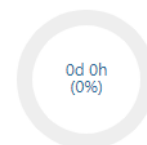
Statements
343



comment, space
ncloc

Technical Debt ?

Total(Technical Debt Ratio)



Issues(Inspections)
0d 0h(0%)

Duplication
0d 0h(0%)

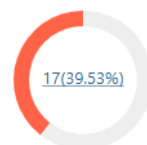
Complexity
0d 0h(0%)

Package Stability
0d 0h(0%)

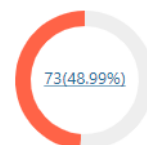
issues duplication complexity stability

Unused Code ? Excel Download

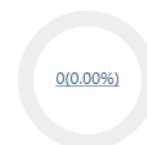
Class



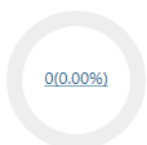
Method



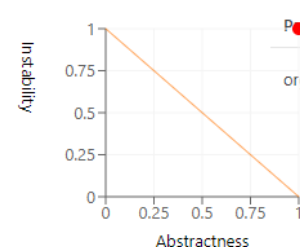
Field



Const



Martin Metrics ?



Package Name

Afferent Coupling

Efferent Coupling

Abstractness

Instability

org.springframework.samples.petclinic.repository.springdatajpa

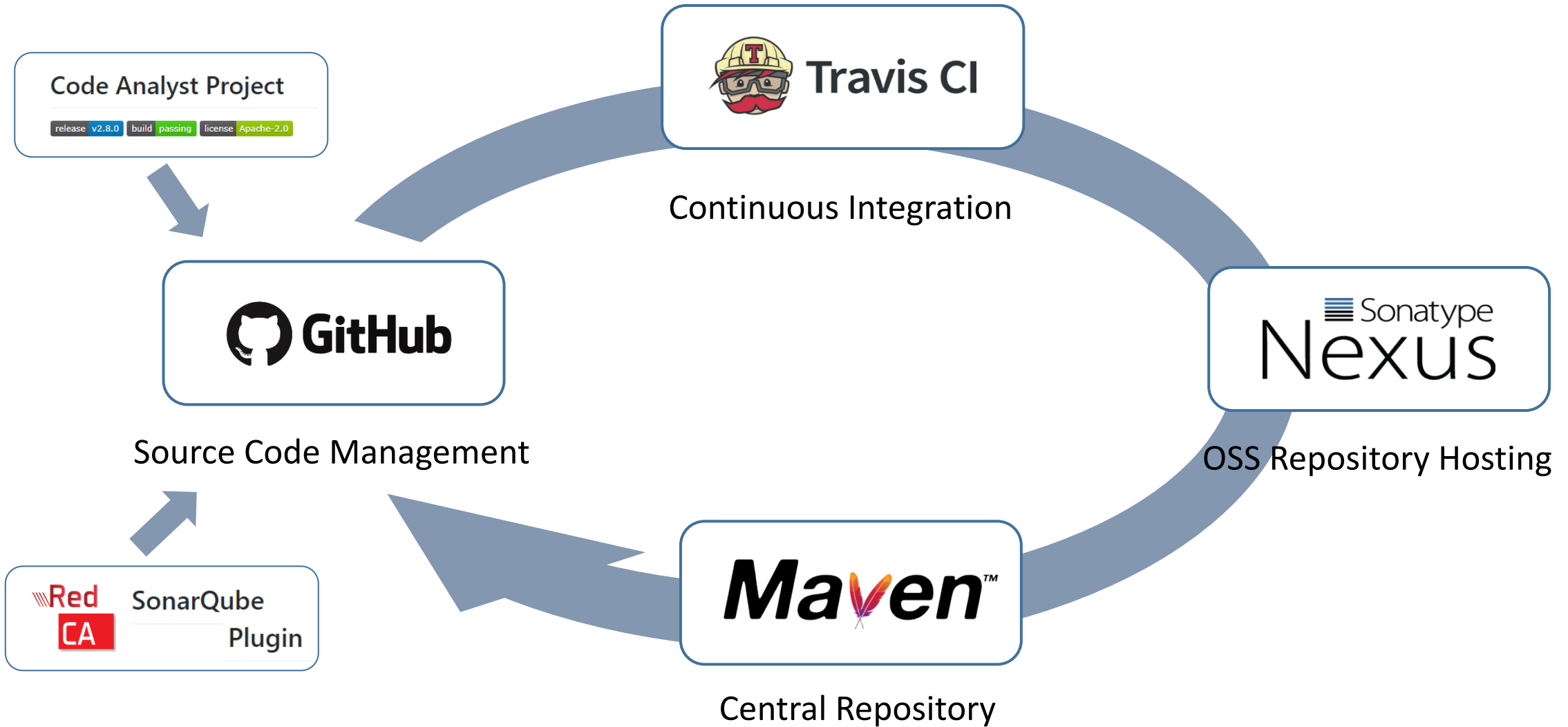
0

7

1

1

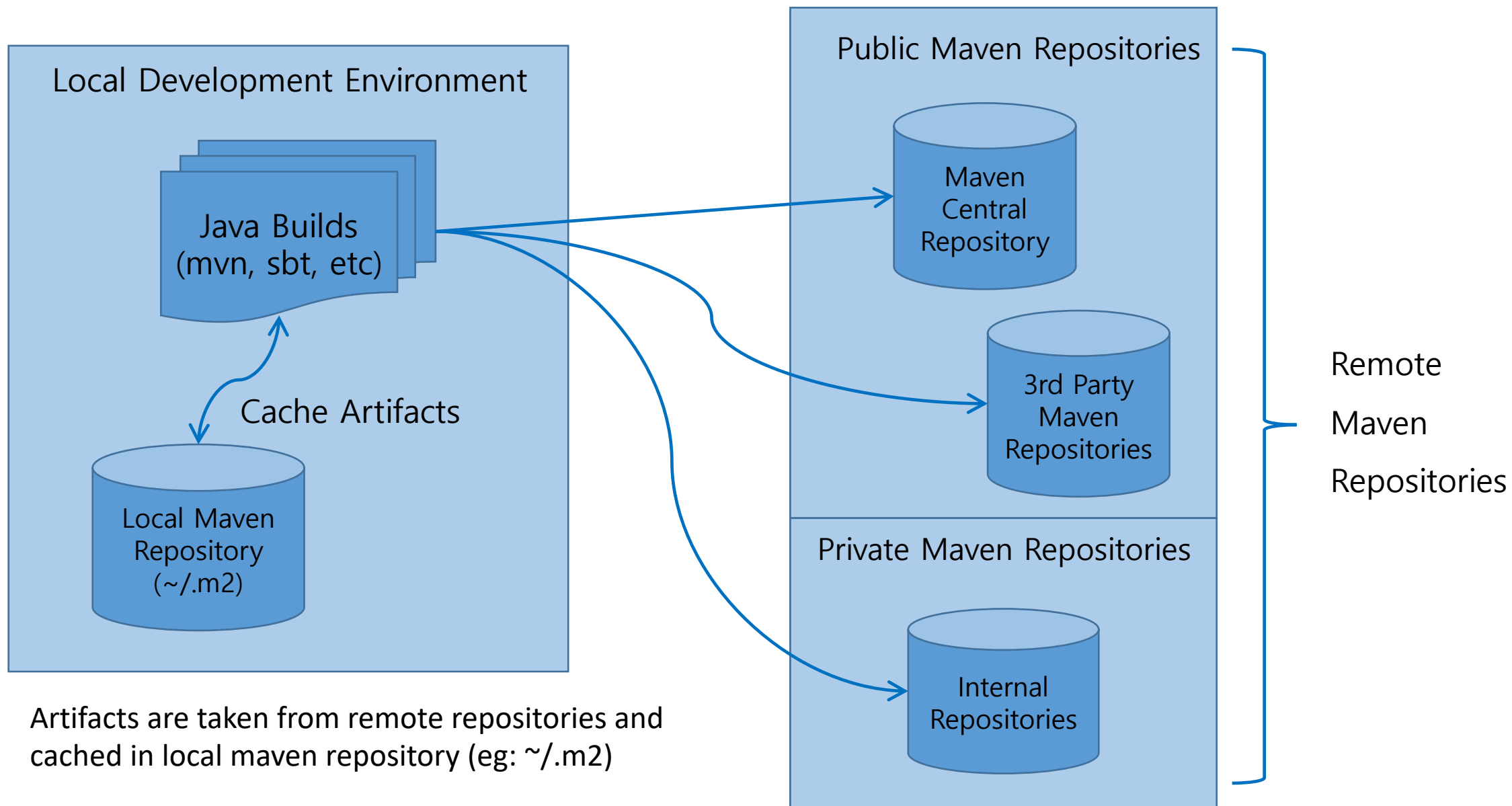
Code Analyst - configuration



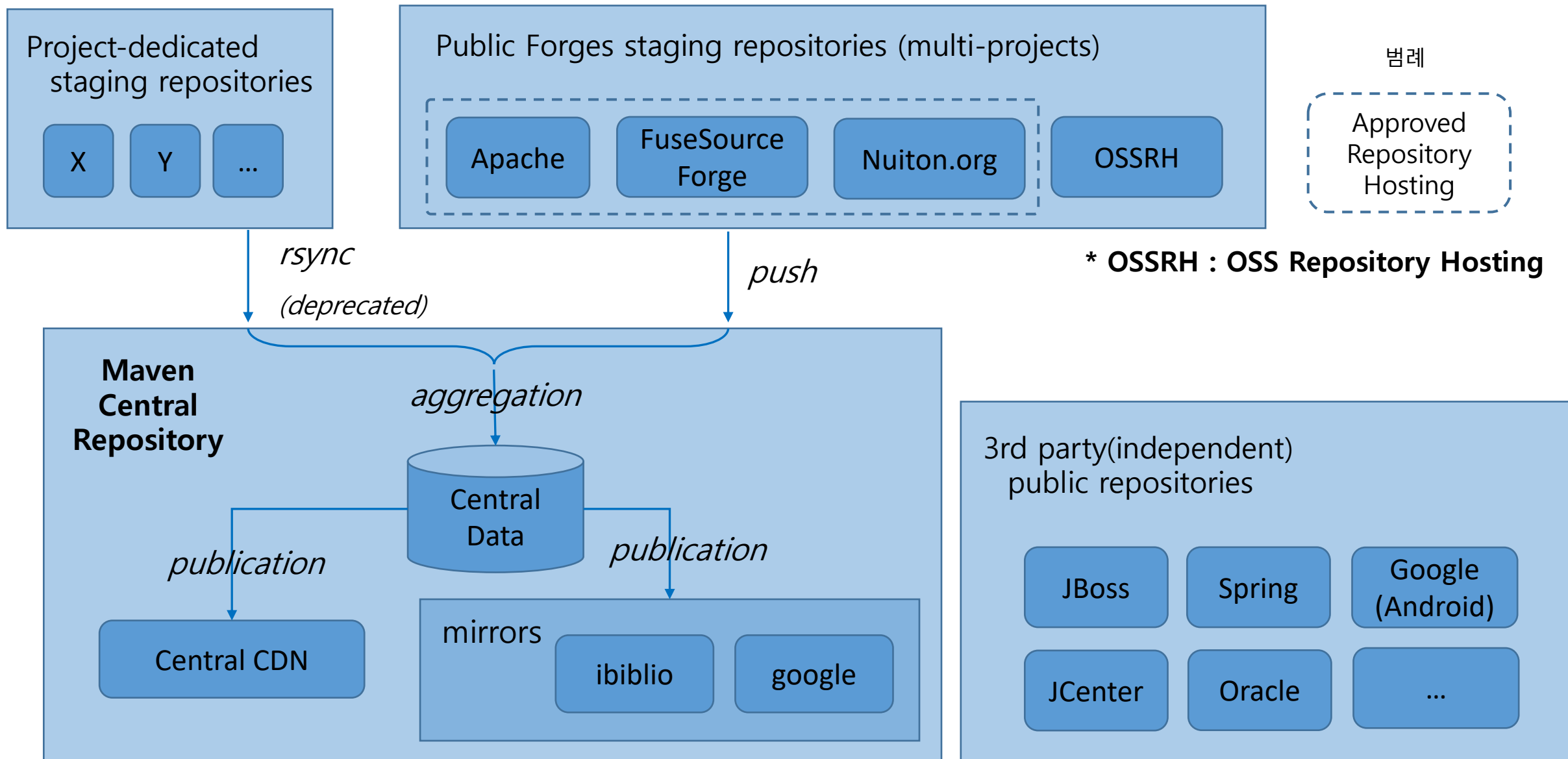
A large, irregular blue ink splash or watercolor blotch serves as the background for the text. The splash is centered and has a textured, painterly appearance with various shades of blue and some white highlights. The text is centered within this splash.

Maven Central Repository 개요

Maven Repositories



Maven Central Repository



Maven Central Repository – cont'd

- URL : <https://repo1.maven.org/maven2/>
- 2019년 현재 약 4M Artifacts(GAV), 300K Unique Artifacts(GA) 제공
- Artifacts 검색 : <https://search.maven.org/> 또는 MvnRepository (<https://mvnrepository.com/>, 1K 이상 repositories 통합 검색 및 부가 정보 제공)
-  **sonatype** 에서 운영
 - Apache Maven 프로젝트 핵심 Contributor
 - Repository Manager 소프트웨어 Nexus 제공 (Open Source / Commercial ver.)
 - OSSRH 서비스도 제공

Indexed Repositories (1192)	
Central	
Sonatype	
Spring Plugins	
Spring Lib M	
Hortonworks	
Atlassian	
JCenter	
JBossEA	
JBoss Releases	
Spring Lib Release	

<MvnRepository>

Requirements for artifacts upload to the central

- Releases : 릴리즈 버전 (삭제 및 변경 불가)
- Javadoc & sources : IDE 연계 지원
- PGP signature : 변조 방지 등
- Minimum POM information : 최소 정보 (license, developers, scm 등)
- Coordinates : groupId 정책 (domain 소유 등)
 - groupId 가이드라인 eg) samsungsds.com -> com.samsungsds,
github.com/*username* -> io.github.*username*
 - artifactId 가이드라인 eg) maven-core, commons-math
 - version 가이드 : Semantic Versioning (참고 : <https://semver.org/lang/ko/>)

A large, irregular blue ink splash or watercolor blotch serves as the background for the text. The splash is centered and has a textured, painterly appearance with various shades of blue and white. The text is white and positioned in the center of the splash.

OSS Repository Hosting 활용

OSSRH overview

- Sonatype Nexus Repository Manager (<https://oss.sonatype.org/>)
- 오픈소스 프로젝트 바이너리 호스팅 서비스 (무료)
- Maven repository 표준 지원 및 다음과 같은 서비스 제공
 - 개발 버전(snapshots) 배포
 - 최종 릴리즈 버전 스테이징
 - 릴리즈 승인 및 Maven Central Repository 동기화

Overall flow

관리적 프로세스

Sonatype
가입



Project 정보
생성 요청



GroupId(domain)
확인 후 승인

승인 후

진행 가능

기술적 프로세스

Javadoc &
Sources 생성



PGP/GPG
전자서명



Distribution
설정



Staging
Repository
배포



“First”
릴리즈



CI 적용



- JIRA 가입 (<https://issues.sonatype.org/secure/Signup!default.jspa>)

Sign up


Email*

Full name*

Username*

Password*

Please enter the word as shown below

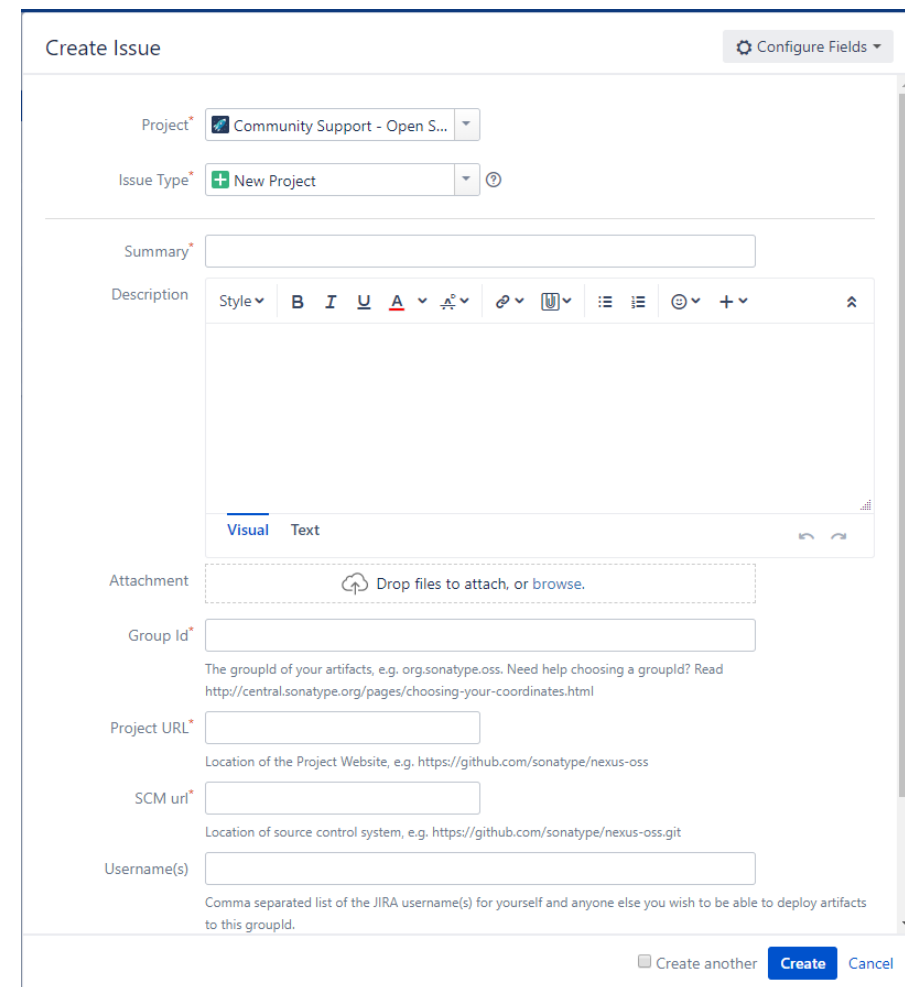


Sign up


Cancel

- JIRA Issue 생성

- Issue Type : New Project
- Summary : 프로젝트 또는 라이브러리 이름
- Description & Attachment : 상세 설명 및 관련 파일
- Group Id : 사용하고자 하는 GroupId
- Project URL : 프로젝트 사이트 주소
- SCM URL : 형상서버 주소
- Username(s) : JIRA 계정 (여러 명 지정 가능)

A screenshot of the 'Create Issue' form in JIRA. The form is titled 'Create Issue' with a 'Configure Fields' link in the top right. It contains several input fields: 'Project' (set to 'Community Support - Open S...'), 'Issue Type' (set to '+ New Project'), 'Summary' (empty), 'Description' (with a rich text editor toolbar), 'Attachment' (with a 'Drop files to attach, or browse.' prompt), 'Group Id' (empty, with a help link), 'Project URL' (empty, with a help link), 'SCM url' (empty, with a help link), and 'Username(s)' (empty, with a help link). At the bottom right, there are three buttons: 'Create another', 'Create', and 'Cancel'.

- Issue Comments 확인 및 조치

✓  Central OSSRH added a comment - 11/27/18 02:28 AM

Do you own the domain samsungsds.com? If so, please verify ownership via one of the following methods:

- Add a TXT record to your DNS referencing this issue (Fastest)
- Setup a redirect to your Github page (if it does not already exist)
- Send an email to central@sonatype.com referencing this issue from a samsungsds.com email address

If you do not own this domain, please read:

<http://central.sonatype.org/pages/choosing-your-coordinates.html>

You may also choose a groupId that reflects your project hosting, in this case, something like io.github.redca-family or com.github.redca-family

- DNS 상에 TXT record 등록 (Issue 참고 URL 등)
- 홈페이지 Redirect 지정 (해당 도메인 주소)
- 도메인 주소 사용 이메일로 메일 전송 (to central@sonatype.com)
- “io.github.[username]” or “com.github.[username]” 요청 시 “TICKET ID” Public Repo 생성 확인

- **Phase** : 최소 수행 단위
 - eg: `compile`, `test`, `package`, `clean`, ...
 - **Lifecycle** : 관련 있는 phase들의 순서가 있는 단계
 - **default** : 빌드 수행 (23개 phases로 구성, `validate` → .. → `generate-sources` → .. → `compile` → .. → `test` → .. → `package` → .. → `verify` → `install` → `deploy`)
 - **clean** : 생성된 파일 삭제 (3개 phases로 구성, `pre-clean` → `clean` → `post-clean`)
 - **site** : 문서 생성 (4개 phases로 구성, `pre-site` → `site` → `post-site` → `site-deploy`)
- ※ `mvn <phase>` : lifecycle 상 앞에 있는 모든 phase들을 순서대로 모두 실행 (특정 phase만 실행 불가)

- **Goal** : plugin의 단위 기능
 - Phase들은 "plugin:goal"에 지정됨
 - eg: `compile` → `compiler:compile`, `test-compile` → `compiler:testCompile`, `test` → `surefire:test`, `package` → `jar:jar` or `war:war`, ...
- Phase가 없는 goal 실행 방법 : `mvn <plugin>:<goal>`
 - eg: `mvn eclipse:eclipse`
- Goal이 없는 phase? eg: `validate`, `initialize`, `verify`, ...

※ 참고 : <http://maven.apache.org/guides/introduction/introduction-to-the-lifecycle.html>

- Maven Javadoc plugin
 - Goal : jar → creates an archive file of the generated Javadocs
 - Phase : package

Tip) "-Xdoclint:none" : "Doc Lint" 점검 해제

```
<plugin>
  <artifactId>maven-javadoc-plugin</artifactId>
  <version>3.0.1</version>
  <configuration>
    <additionalJOption>-Xdoclint:none</additionalJOption>
  </configuration>
  <executions>
    <execution>
      <id>attach-javadocs</id>
      <goals>
        <goal>jar</goal>
      </goals>
    </execution>
  </executions>
</plugin>
```


- Maven Source plugin

```
<plugin>
  <artifactId>maven-source-plugin</artifactId>
  <version>3.0.1</version>
  <executions>
    <execution>
      <id>attach-sources</id>
      <goals>
        <goal>jar-no-fork</goal>
      </goals>
    </execution>
  </executions>
</plugin>
```

- Goal : jar-no-fork → bundle the main sources of the project into a jar archive (without fork)
- Phase : package

Tip) "jar"가 아닌 "jar-no-fork" goal 사용 : 빌드 라이프사이클 상에 포함

- PGP(Pretty Good Privacy) 전자서명 필요

- GnuPG(GPG, GNU Privacy Guard) 또는 Gpg4Win 사용



- OpenPGP(RFC4880) 표준 지원

- 공개키 기반으로 Public Key에 대한 공개 필요 (전자서명 검증 또는 암호화 전송)

- 절차 개요

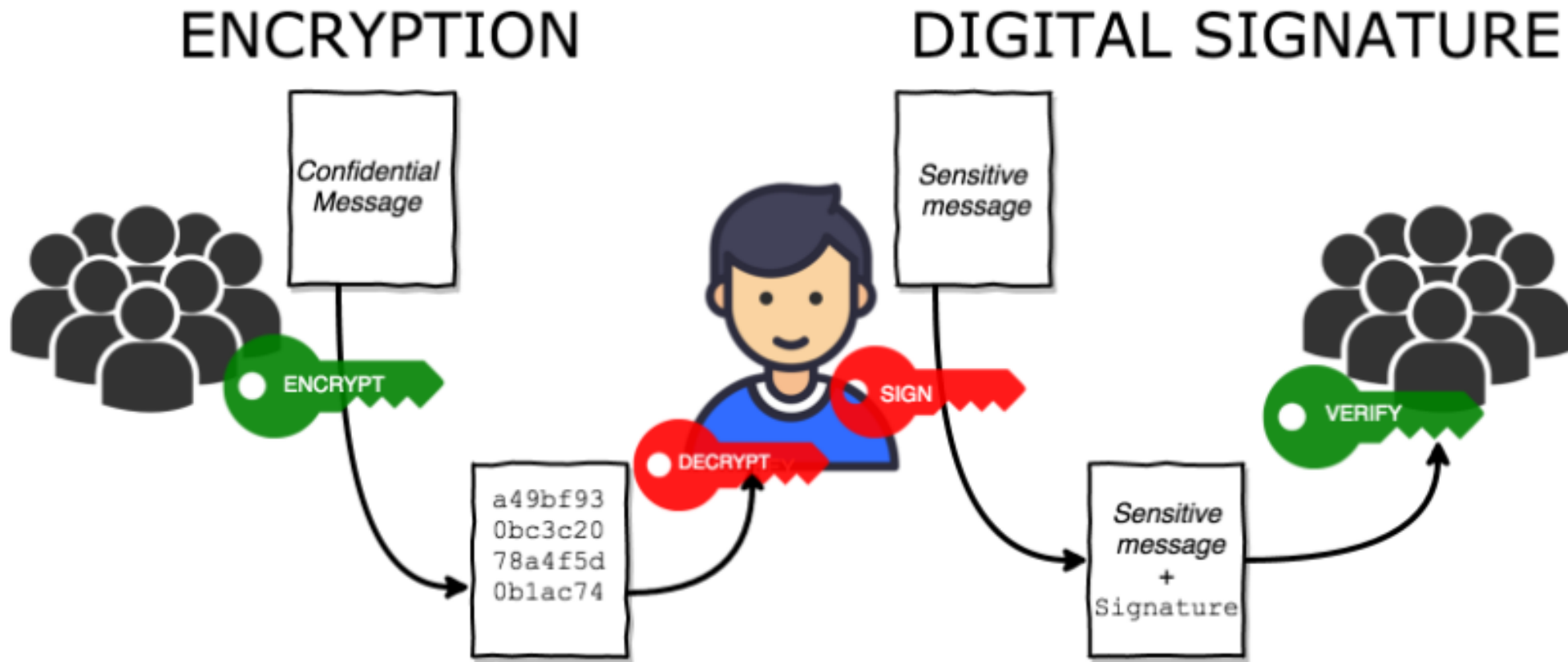
Keys(Public/Secret) 생성

전자서명 및 확인

Public Key 공개

빌드 환경 적용

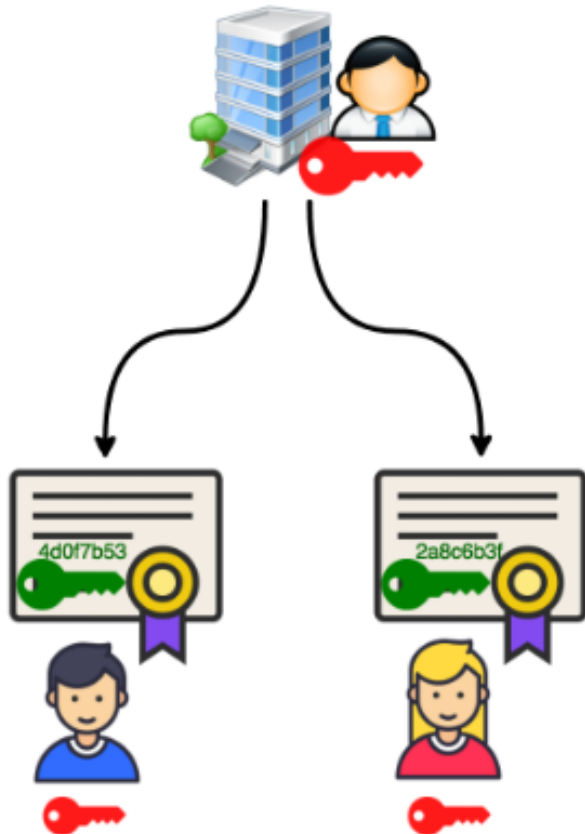
[참고] Encryption & Digital Signature



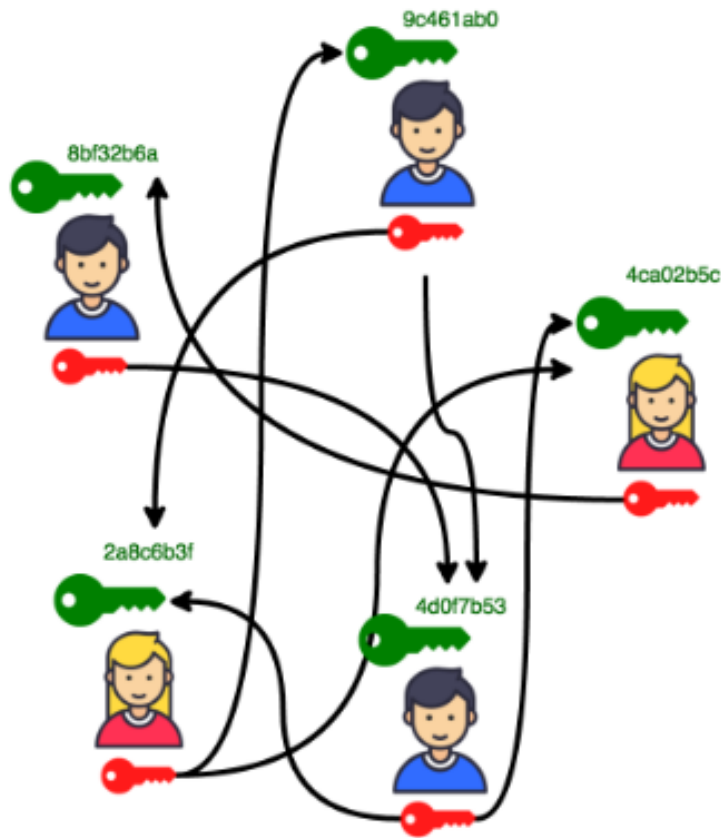
<이미지 출처: Jeroen Ooms, *Encryption and Digital Signatures using GPG*
(<https://cran.r-project.org/web/packages/gpg/vignettes/intro.html>)>

[참고] PKI vs PGP

CERTIFICATE AUTHORITY



PGP / WEB OF TRUST

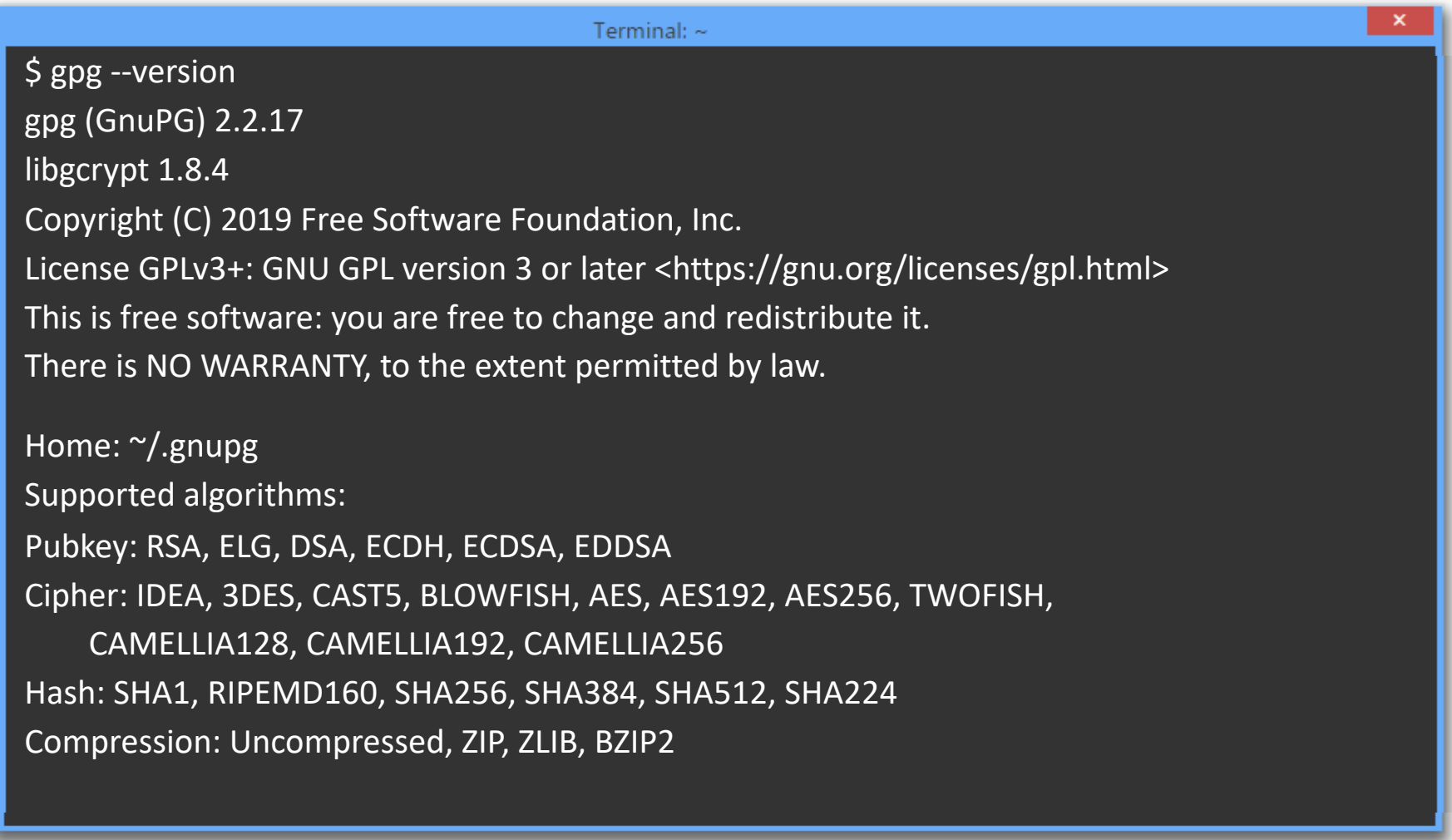


With PGP KeyServer

- <https://pgp.mit.edu>
- <https://keyserver.ubuntu.com>
- <https://sks-keyservers.net/> 등

<이미지 출처: Jeroen Ooms, *Encryption and Digital Signatures using GPG*
(<https://cran.r-project.org/web/packages/gpg/vignettes/intro.html>)>

<0. GPG 설치 및 확인>

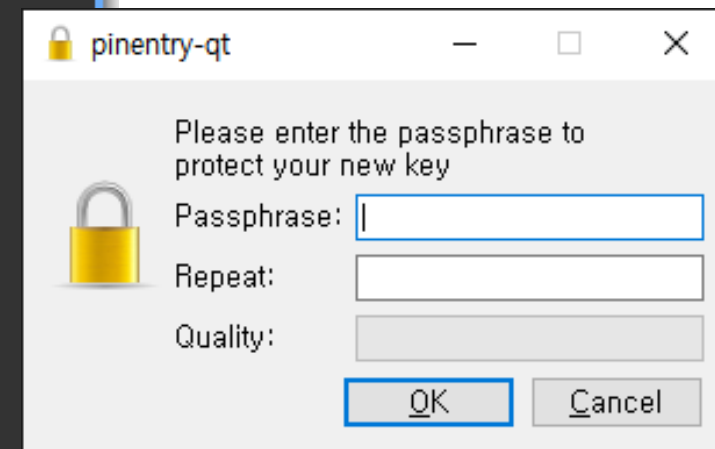
A terminal window with a blue title bar labeled "Terminal: ~" and a red close button. The terminal output shows the command "\$ gpg --version" and its output, which includes the version number (2.2.17), copyright information (2019 Free Software Foundation, Inc.), license (GPLv3+), and a list of supported algorithms for public keys, ciphers, hashes, and compression.

```
$ gpg --version
gpg (GnuPG) 2.2.17
libgcrypt 1.8.4
Copyright (C) 2019 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <https://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Home: ~/.gnupg
Supported algorithms:
Pubkey: RSA, ELG, DSA, ECDH, ECDSA, EDDSA
Cipher: IDEA, 3DES, CAST5, BLOWFISH, AES, AES192, AES256, TWOFISH,
        CAMELLIA128, CAMELLIA192, CAMELLIA256
Hash: SHA1, RIPEMD160, SHA256, SHA384, SHA512, SHA224
Compression: Uncompressed, ZIP, ZLIB, BZIP2
```

<1. Key Pair 생성>

```
Terminal: ~  
$ gpg --gen-key  
gpg (GnuPG) 2.2.17; Copyright (C) 2019 Free Software Foundation, Inc.  
This is free software: you are free to change and redistribute it.  
There is NO WARRANTY, to the extent permitted by law.  
GnuPG needs to construct a user ID to identify your key.  
  
Real name: Vincent Han  
Email address: switchover@gmail.com  
You selected this USER-ID:  
  "Vincent Han <switchover@gmail.com>"  
  
Change (N)ame, (E)mail, or (O)kay/(Q)uit? ○  
We need to generate a lot of random bytes. It is a good idea to perform  
some other action (type on the keyboard, move the mouse, utilize the  
disks) during the prime generation; this gives the random number  
generator a better chance to gain enough entropy.
```



<1. Key Pair 생성 – cont'd>

Terminal: ~

We need to generate a lot of random bytes. It is a good idea to perform some other action (type on the keyboard, move the mouse, utilize the disks) during the prime generation; this gives the random number generator a better chance to gain enough entropy.

gpg: key FD76895355C6092D marked as ultimately trusted

gpg: revocation certificate stored as 'C:/Users/SDS/AppData/Roaming/gnupg/openpgp-revocs.d\0B39DBE946ADB31AD2B5B369FD76895355C6092D.rev'

public and secret key created and signed.

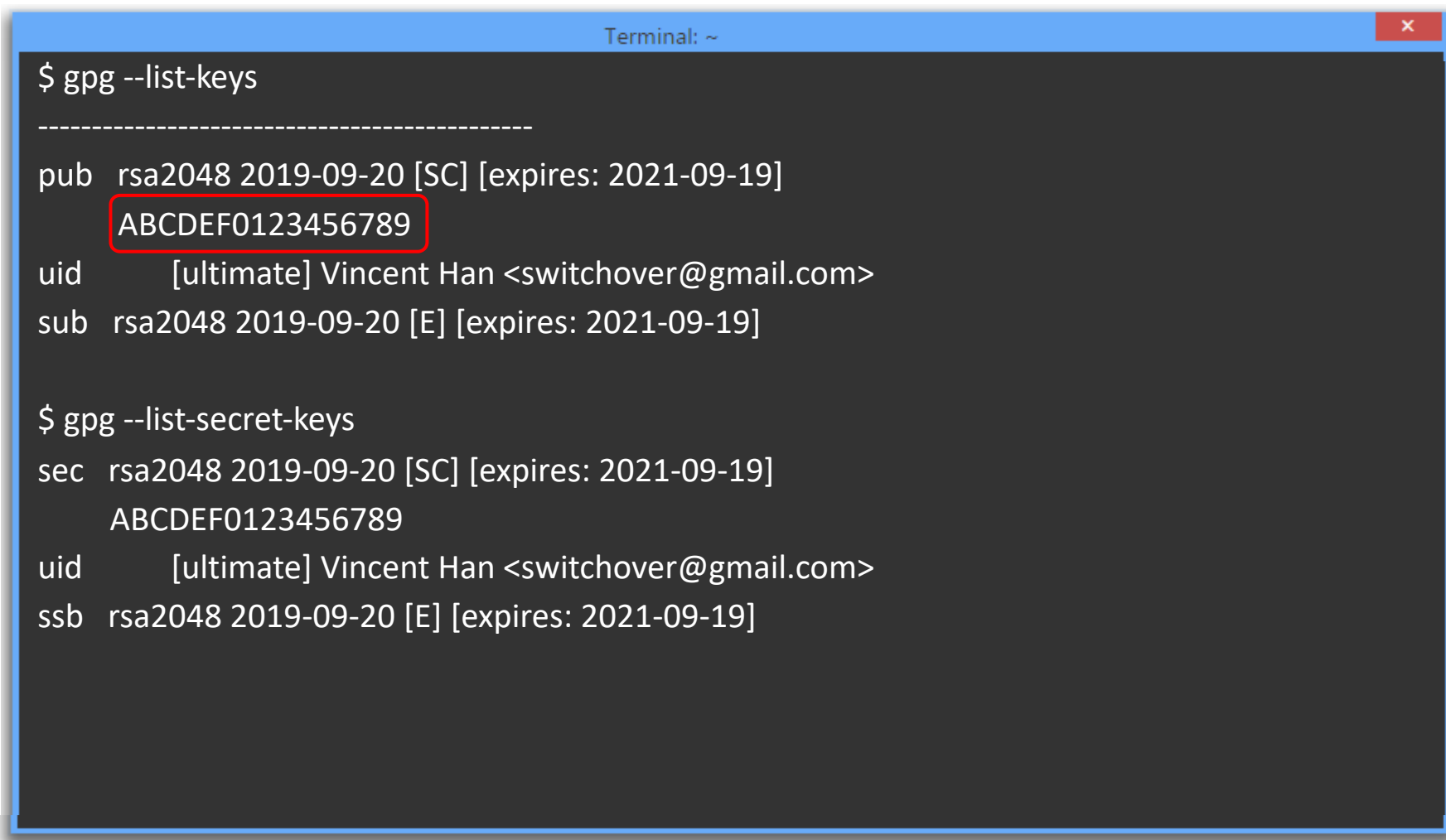
pub rsa2048 2019-09-20 [SC] [expires: 2021-09-19]

ABCDEF0123456789

uid Vincent Han <switchover@gmail.com>

sub rsa2048 2019-09-20 [E] [expires: 2021-09-19]

<2. 전자서명 및 확인>

A terminal window titled "Terminal: ~" with a blue header bar and a red close button. It displays the output of two GPG commands. The first command, "\$ gpg --list-keys", shows a public key for Vincent Han with ID ABCDEF0123456789, which is highlighted with a red rectangle. The second command, "\$ gpg --list-secret-keys", shows the corresponding secret key with the same ID. The output for both commands includes details like key type (rsa2048), creation date (2019-09-20), and expiration date (2021-09-19).

```
Terminal: ~
$ gpg --list-keys
-----
pub  rsa2048 2019-09-20 [SC] [expires: 2021-09-19]
    ABCDEF0123456789
uid      [ultimate] Vincent Han <switchover@gmail.com>
sub  rsa2048 2019-09-20 [E] [expires: 2021-09-19]

$ gpg --list-secret-keys
sec  rsa2048 2019-09-20 [SC] [expires: 2021-09-19]
    ABCDEF0123456789
uid      [ultimate] Vincent Han <switchover@gmail.com>
ssb  rsa2048 2019-09-20 [E] [expires: 2021-09-19]
```

<2. 전자서명 및 확인 – cont'd>

```
$ gpg -ab test.txt
```

```
$ ls test.txt.*
```

```
test.txt  test.txt.asc
```

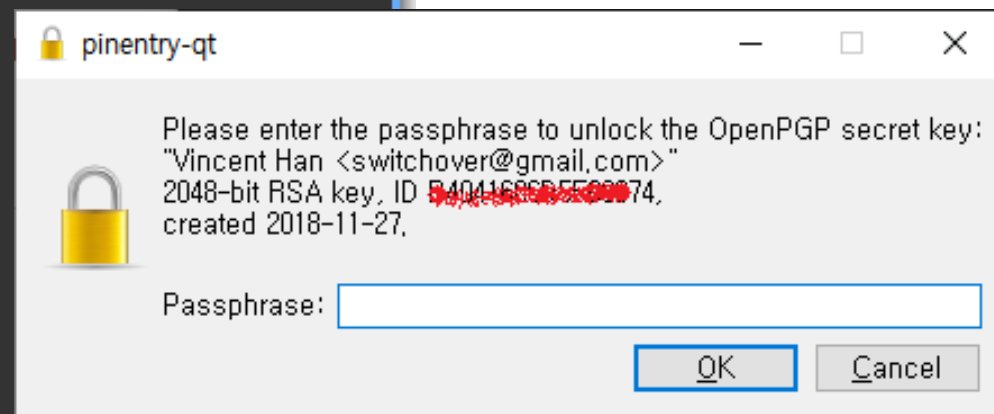
```
$ gpg --verify test.txt.asc
```

```
gpg: assuming signed data in 'test.txt'
```

```
gpg: Signature made 09/20/19 14:07:34 대한민국 표준시
```

```
gpg:      using RSA key ABCDEF0123456789
```

```
gpg: Good signature from "Vincent Han <switchover@gmail.com>" [ultimate]
```



<3. Public Key 공개 및 검색>

The image displays two overlapping windows. The background window is a terminal titled "Terminal: ~" with a black background and white text. It shows the command: `$ gpg --keyserver hkp://pool.sks-keyservers.net --send-keys ABCDEF0123456789`. The foreground window is a web browser with a colorful, abstract background. The address bar shows "OpenPGP Keyserver" and "keys.gnupg.net". The main content area of the browser displays "OpenPGPkeyserver" in large black text. Below this is a search bar with the placeholder text "Search for an OpenPGP Public Key, ie 0x.". There are two blue buttons: "Search Key" with a magnifying glass icon and "Submit Key" with an upload icon. Below these buttons is a link for "Advanced Options". At the bottom of the browser window, there are links for "about" and "statistics", and a footer note: "Provided as a public service by Daniel Austin."

<4. Maven 적용>

- Maven GPG plugin

```
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-gpg-plugin</artifactId>
  <version>1.6</version>
  <executions>
    <execution>
      <id>sign-artifacts</id>
      <phase>verify</phase>
      <goals>
        <goal>sign</goal>
      </goals>
    </execution>
  </executions>
</plugin>
```

- Goal : sign
- Phase : verify

```
<profiles>
  <profile>
    <properties>
      <gpg.executable>gpg</gpg.executable>
      <gpg.passphrase>${env.MAVEN_GPG_PASSPHRASE}</gpg.passphrase>
    </properties>
  </profile>
</profiles>
```

➔ settings.xml 설정 (환경 변수 지정 또는 직접 지정)

- Sonatype 설정 완료 comments 확인

✓  Terry Yanko added a comment - 11/27/18 02:07 PM

Configuration has been prepared, now you can:

- Deploy snapshot artifacts into repository <https://oss.sonatype.org/content/repositories/snapshots>
- Deploy release artifacts into the staging repository <https://oss.sonatype.org/service/local/staging/deploy/maven2>
- Promote staged artifacts into repository 'Releases'
- Download snapshot and release artifacts from group <https://oss.sonatype.org/content/groups/public>
- Download snapshot, release and staged artifacts from staging group <https://oss.sonatype.org/content/groups/staging>

please comment on this ticket when you promoted your first release, thanks

- Maven Distribution Management 설정 (#1 Maven deploy plugin)

```
<distributionManagement>
  <snapshotRepository>
    <id>ossrh</id>
    <url>https://oss.sonatype.org/content/repositories/snapshots</url>
  </snapshotRepository>
  <repository>
    <id>ossrh</id>
    <url>https://oss.sonatype.org/service/local/staging/deploy/maven2</url>
  </repository>
</distributionManagement>
```

- Phase : deploy

```
<servers>
  <server>
    <id>ossrh</id>
    <username>${env.MAVEN_REPO_USERNAME}</username>
    <password>${env.MAVEN_REPO_PASSWORD}</password>
  </server>
</servers>
```

➔ settings.xml 설정 (환경 변수 지정 또는 직접 지정)

- Maven Distribution Management 설정 (#2 Nexus Staging Maven Plugin)

```
<plugin>
  <groupId>org.sonatype.plugins</groupId>
  <artifactId>nexus-staging-maven-plugin</artifactId>
  <version>1.6.8</version>
  <extensions>true</extensions>
  <configuration>
    <serverId>ossrh</serverId>
    <nexusUrl>https://oss.sonatype.org/</nexusUrl>
    <autoReleaseAfterClose>true</autoReleaseAfterClose>
  </configuration>
</plugin>
```

- Phase : verify

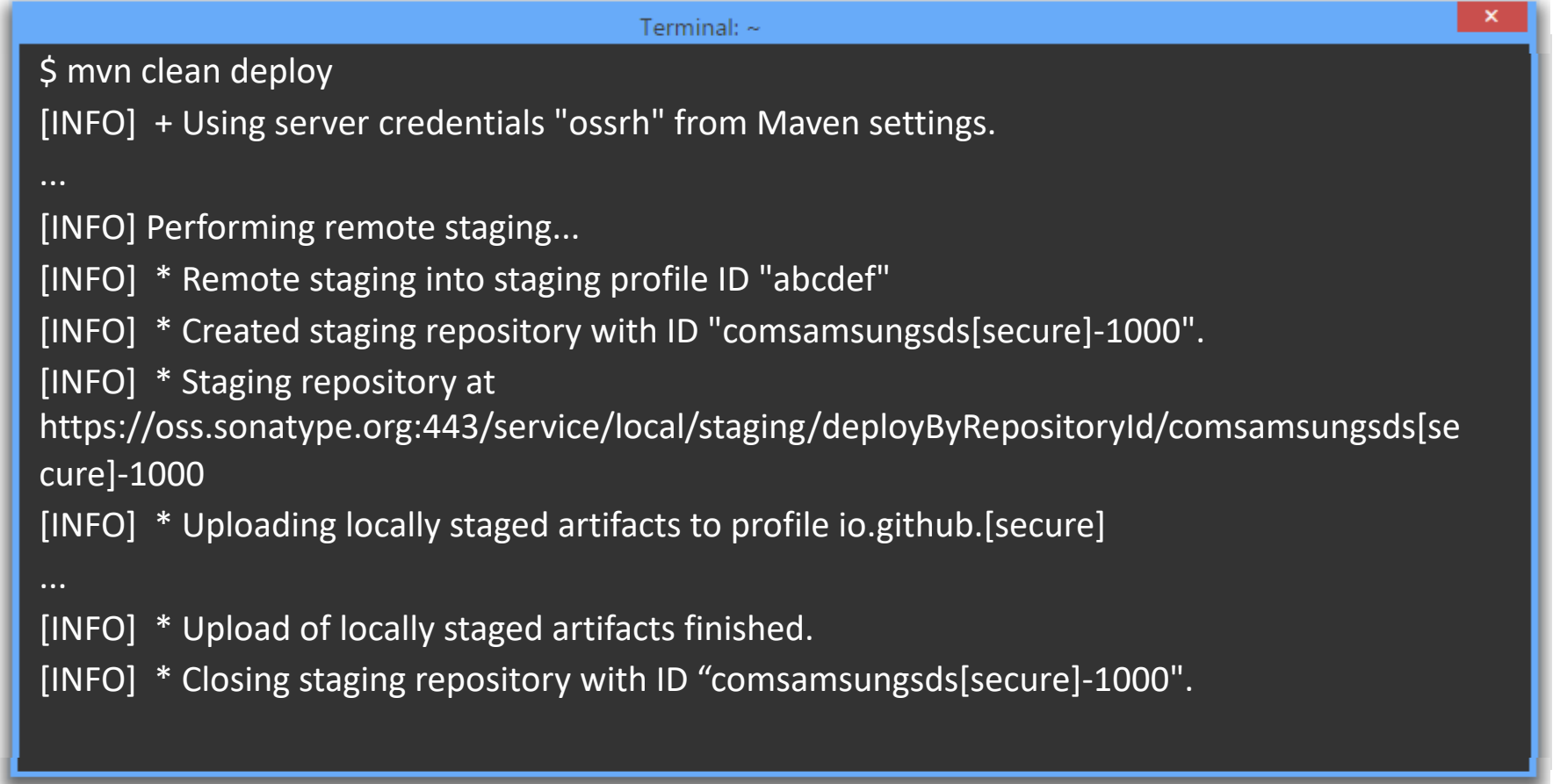
```
<servers>
  <server>
    <id>ossrh</id>
    <username>${env.MAVEN_REPO_USERNAME}</username>
    <password>${env.MAVEN_REPO_PASSWORD}</password>
  </server>
</servers>
```

Tip) Nexus Staging Maven Plugin 사용 권고

➔ 일부 CI 환경에서 문제 발생 (Artifact들이 push될 때마다 분산 처리에 의해 IP가 달라지며, 이에 따라 artifact들이 분산 등록됨)

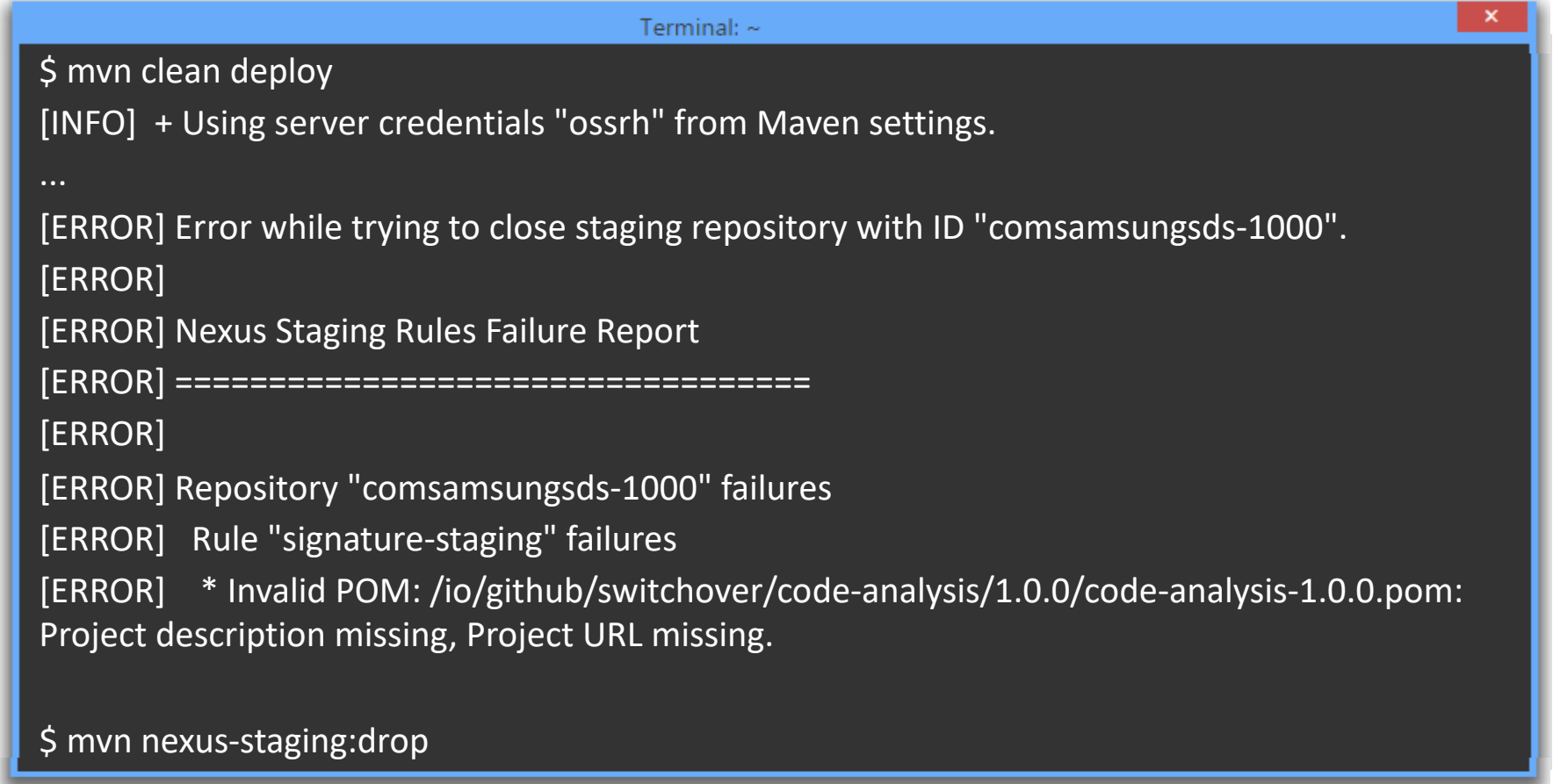
➔ settings.xml 설정
(환경 변수 지정 또는 직접 지정)

- deploy 실행 (성공 시)

A terminal window with a blue title bar labeled "Terminal: ~" and a red close button. The terminal displays the output of a Maven command. The text is as follows:

```
$ mvn clean deploy
[INFO] + Using server credentials "ossrh" from Maven settings.
...
[INFO] Performing remote staging...
[INFO] * Remote staging into staging profile ID "abcdef"
[INFO] * Created staging repository with ID "comsamsungsds[secure]-1000".
[INFO] * Staging repository at
https://oss.sonatype.org:443/service/local/staging/deployByRepositoryId/comsamsungsds[se
cure]-1000
[INFO] * Uploading locally staged artifacts to profile io.github.[secure]
...
[INFO] * Upload of locally staged artifacts finished.
[INFO] * Closing staging repository with ID "comsamsungsds[secure]-1000".
```

- deploy 실행 (실패 시)

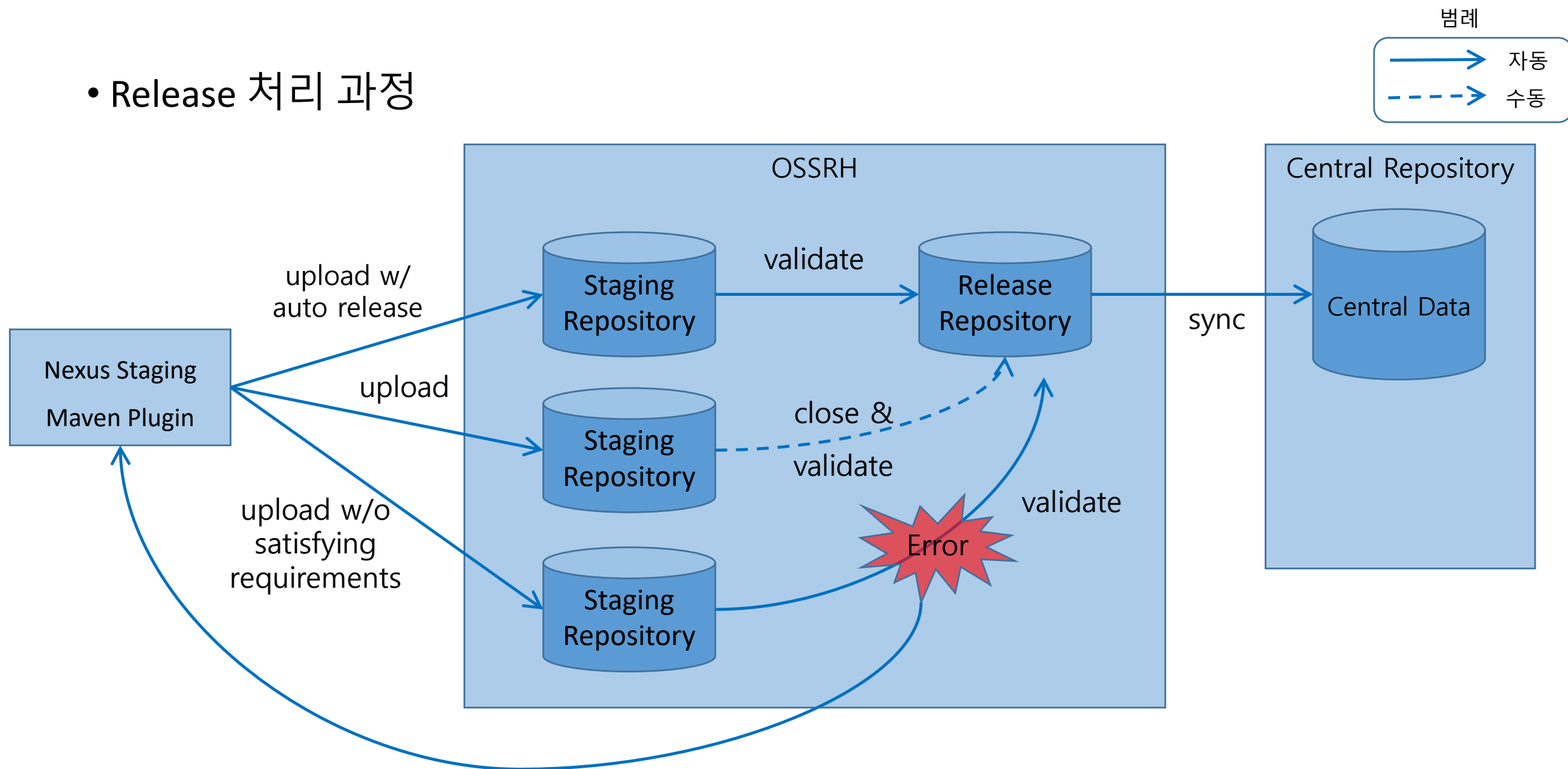
A terminal window with a blue title bar labeled "Terminal: ~" and a red close button. The terminal output shows a Maven clean deploy command followed by several error messages. The errors indicate a failure to close the staging repository and a Nexus Staging Rules Failure Report. The report details failures for the repository "comsamsungsds-1000", specifically for the rule "signature-staging", citing an invalid POM for the project at "/io/github/switchover/code-analysis/1.0.0/code-analysis-1.0.0.pom" due to missing project description and URL. The final command shown is "mvn nexus-staging:drop".

```
$ mvn clean deploy
[INFO] + Using server credentials "ossrh" from Maven settings.
...
[ERROR] Error while trying to close staging repository with ID "comsamsungsds-1000".
[ERROR]
[ERROR] Nexus Staging Rules Failure Report
[ERROR] =====
[ERROR]
[ERROR] Repository "comsamsungsds-1000" failures
[ERROR] Rule "signature-staging" failures
[ERROR] * Invalid POM: /io/github/switchover/code-analysis/1.0.0/code-analysis-1.0.0.pom:
Project description missing, Project URL missing.

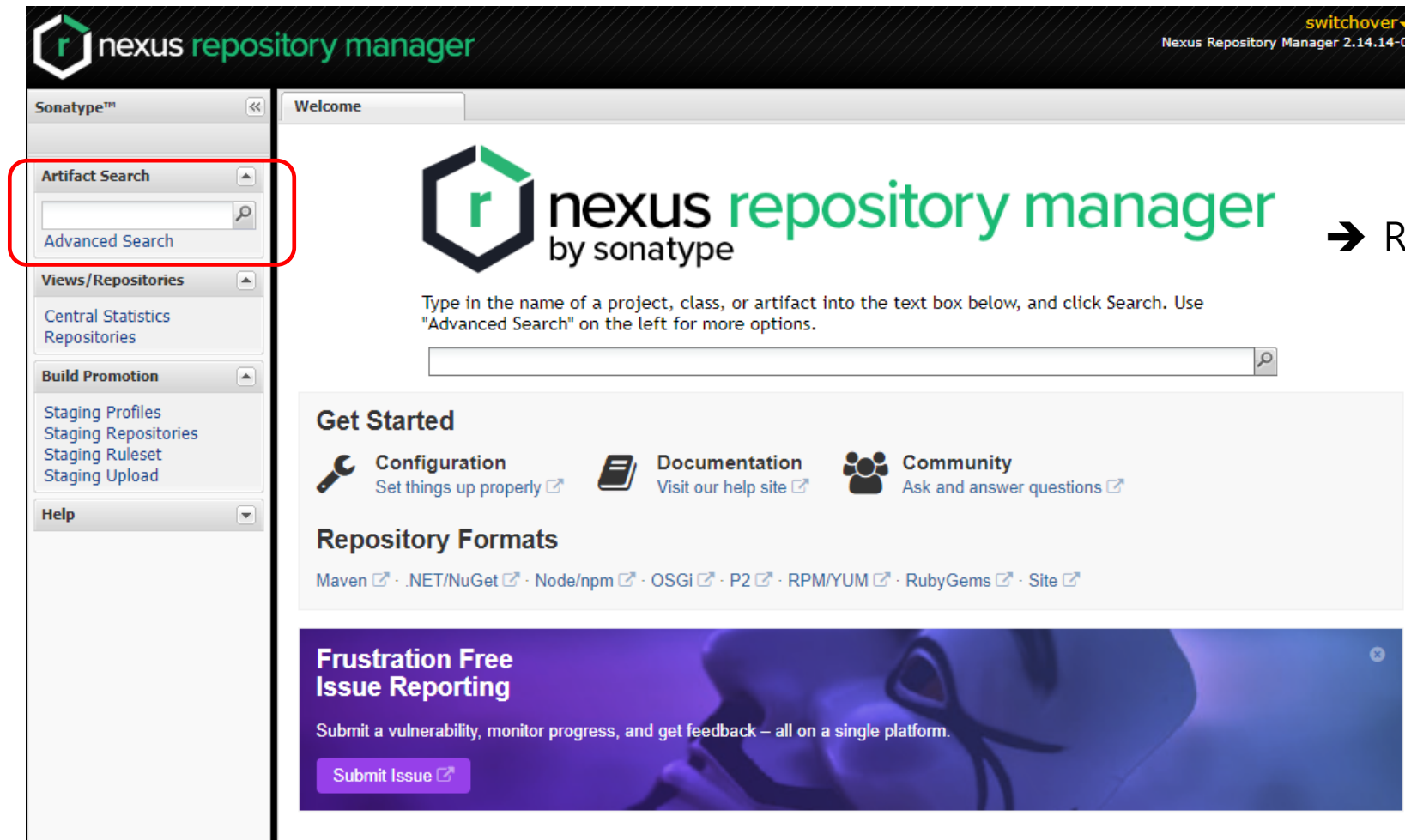
$ mvn nexus-staging:drop
```

➔ 오류 artifact 삭제

• Release 처리 과정



- Artifacts 확인 (<https://oss.sonatype.org/>)



The screenshot displays the Nexus Repository Manager web interface. The left sidebar features a navigation menu with the following items: Sonatype™, Artifact Search (highlighted with a red box), Views/Repositories, Build Promotion, and Help. The main content area includes a 'Welcome' message, the Nexus logo, a search bar, and sections for 'Get Started' (Configuration, Documentation, Community) and 'Repository Formats' (Maven, .NET/NuGet, Node/npm, OSGi, P2, RPM/YUM, RubyGems, Site). A banner at the bottom promotes 'Frustration Free Issue Reporting'.

→ Release repository 조회

- Artifacts 확인 (<https://oss.sonatype.org/>) – cont'd

The screenshot shows the Sonatype OSSRH Staging Repository search results for the keyword 'com.samsungsds'. The search results table displays two records for the artifact 'code-analyst' in version 2.8.0 and 2.7.0. The table columns include Group, Artifact, Version, Age, Popularity, Security Issues, License Threat, and Download. The download column lists 'pom, jar, sources.jar, javadoc.jar' for both versions.

Group	Artifact	Version	Age	Popularity	Security Issues	License Threat	Download
com.samsungsds...	code-analyst	2.8.0	ⓧ	ⓧ	ⓧ	ⓧ	pom, jar, sources.jar, javadoc.jar
com.samsungsds...	code-analyst	2.7.0	ⓧ	ⓧ	ⓧ	ⓧ	pom, jar, sources.jar, javadoc.jar

Displaying Top 7 records × Clear Results

Refresh | Viewing Repository: Releases

Releases

- com
 - samsungsds
 - analyst
 - code-analyst
 - 2.5.0
 - 2.5.1
 - 2.6.0
 - 2.6.1
 - 2.7.0
 - 2.8.0
 - code-analyst-2.8.0-javadoc.jar
 - code-analyst-2.8.0-sources.jar
 - code-analyst-2.8.0.jar

Maven Artifact Artifact Metadata Archive Browser Maven Dependency Com

Group: com.samsungsds.analyst

Artifact: code-analyst

Version: 2.5.0

Extension: jar

XML:

```
<dependency>  
<groupId>com.samsungsds.analyst</groupId>  
<artifactId>code-analyst</artifactId>  
<version>2.5.0</version>  
</dependency>
```

- Artifacts 확인 (w/o autoReleaseAfterClose)

The screenshot shows the OSSRH Staging Repository interface. On the left, the 'Staging Repositories' link is highlighted in red. The main table lists several staging repositories, with the 'iogithubswitchover-1003' repository highlighted in red. The table columns are Repository, Profile, Status, Updated, Descr..., and IQ Policy Viola.

Repository	Profile	Status	Updated	Descr...	IQ Policy Viola
central_bundles-19100	Central Bundles	closed	2019-Sep-20 05:16:03	de.pe...	No IQ Server c
central_bundles-19131	Central Bundles	closed	2019-Sep-22 23:44:12	Implic...	No IQ Server c
central_bundles-19133	Central Bundles	closed	2019-Sep-23 18:53:07	Implic...	No IQ Server c
central_bundles-19139	Central Bundles	closed	2019-Sep-23 21:51:58	com....	No IQ Server c
central_bundles-19140	Central Bundles	closed	2019-Sep-23 22:22:16	com....	No IQ Server c
central_bundles-19141	Central Bundles	closed	2019-Sep-24 00:19:03	com....	No IQ Server c
central_bundles-19142	Central Bundles	closed	2019-Sep-24 00:35:40	com....	No IQ Server c
iogithubswitchover-1003	io.github.switch...	closed	2019-Sep-24 13:07:43	io.git...	No IQ Server c

→ Staging repository 조회

- Artifacts 확인 (w/o autoReleaseAfterClose) – cont'd

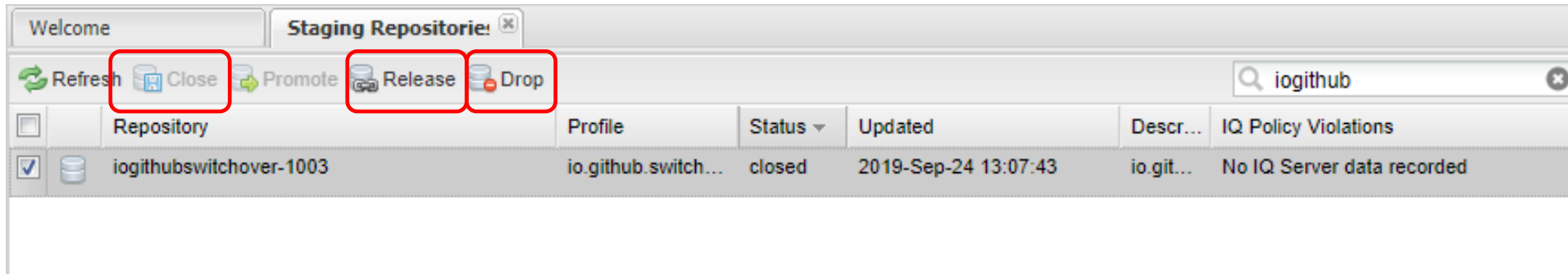
The screenshot displays the OSSRH Staging Repository interface for the repository `io.github.switchover-1003`. The **Content** tab is selected, showing a tree view of the repository structure. A red box highlights the `code-analysis` directory, and a red arrow points from the `Content` tab in the left sidebar to this directory. The artifact list for `code-analysis-1.1.0` is shown, including files like `code-analysis-1.1.0-javadoc.jar`, `code-analysis-1.1.0-javadoc.jar.asc`, `code-analysis-1.1.0-javadoc.jar.md5`, `code-analysis-1.1.0-javadoc.jar.sha1`, `code-analysis-1.1.0-sources.jar`, `code-analysis-1.1.0-sources.jar.asc`, `code-analysis-1.1.0-sources.jar.md5`, `code-analysis-1.1.0-sources.jar.sha1`, `code-analysis-1.1.0.jar`, `code-analysis-1.1.0.jar.asc`, `code-analysis-1.1.0.jar.md5`, `code-analysis-1.1.0.jar.sha1`, `code-analysis-1.1.0.pom`, `code-analysis-1.1.0.pom.asc`, `code-analysis-1.1.0.pom.md5`, and `code-analysis-1.1.0.pom.sha1`. The `maven-metadata.xml` file is also visible at the bottom of the list.

Repository: `io.github.switchover-1003` (`io.github.switchover`)
Created: Tuesday, September 24, 2019 13:06:20 (GMT+0900)
Updated: Tuesday, September 24, 2019 13:07:43 (GMT+0900)
URL: <https://oss.sonatype.org/content/repositories/io.github.switchover-1003>
Activity: Last operation completed successfully
Owner: `switchover` (104.198.131.58)
User-Agent: `Nexus-Client/2.14.3-02`

Description: `io.github.switchover:code-analysis:1.1.0`

→ Content 확인

- Close & Release




- ➔ "Close" : artifacts 필요 요건 점검 (requirements)
- ➔ "Release" : Central Repository와 동기화를 위해 release repository로 릴리즈
- ➔ "Drop" : "Close" 시 오류가 발생하면, 삭제 처리

- Sonatype JIRA issue(ticket) comment 확인 또는 등록

✓  Central OSSRH added a comment - 1 hour ago

➔ 자동 release 적용 시

Central sync is activated for io.github.switchover. After you successfully release, your component will be published to Central, typically within 10 minutes, though updates to search.maven.org can take up to two hours.

✓  Vincent Han added a comment - 12/03/18 12:56 AM

➔ 수동 release 적용 시

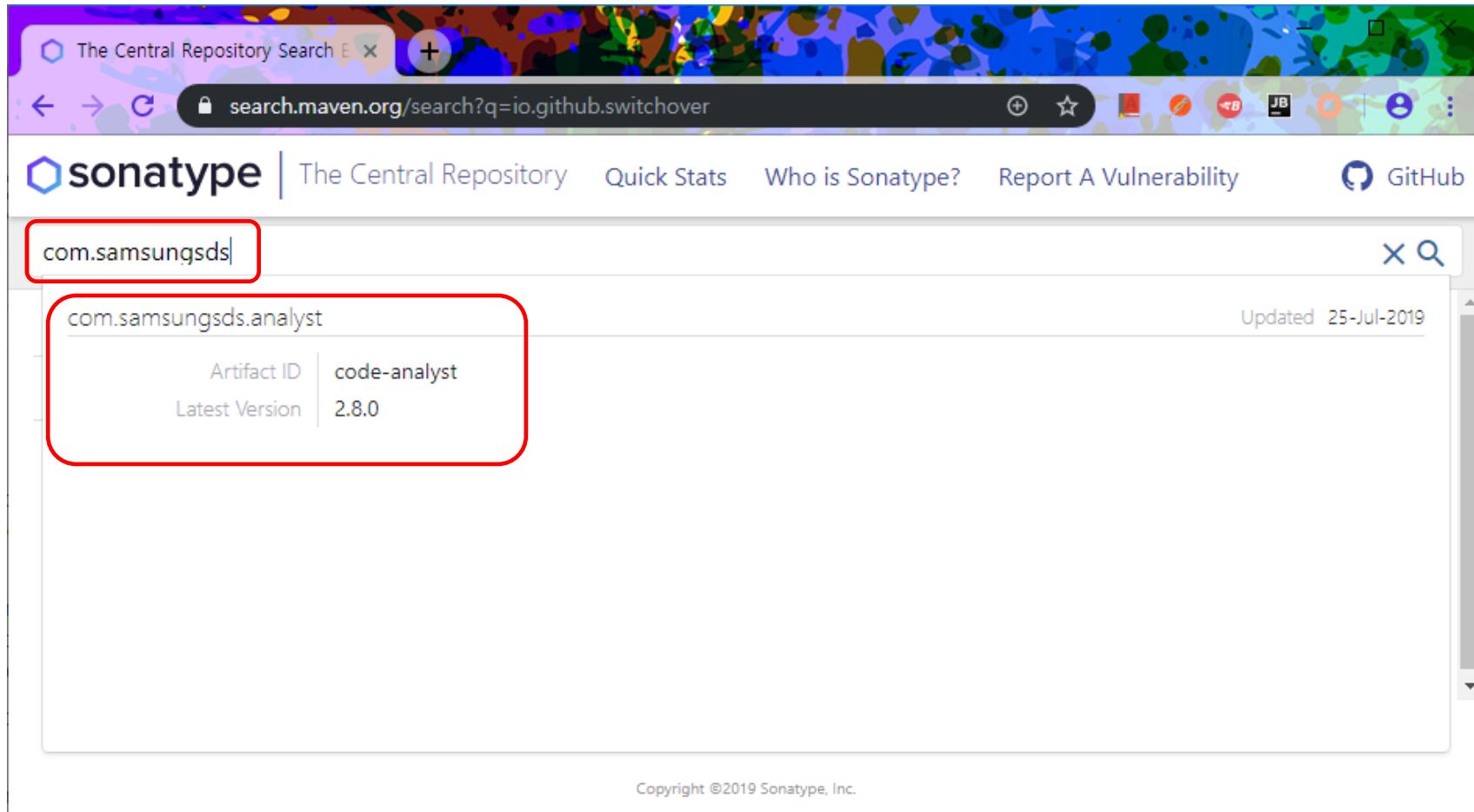
First release is promoted!

(.../com/samsungsds/analyst/code-analyst)

Thank you so much

And have a nice day!!

- Search in Maven Central Repository (<https://search.maven.org>)



➔ 조회되기 까지
10분에서 2시간까지
소요

- CI 적용 시 고려사항
 - 실제 배포가 아닌 경우에도 불필요한 Maven plugin들이 실행됨 (속도 문제)
 - ➔ Maven Profile 사용
 - GPG 보안 키(Secret Key) 대한 패스워드(passphrase) 및 Sonatype 인증 정보 노출 문제
 - ➔ 각 CI 환경에서 제공하는 변수 암호화 적용

- Maven Profile 적용

```
<!-- Profiles -->
<profiles>
  <profile>
    <id>release</id>
    <build>
      <finalName>${project.artifactId}-${project.version}</finalName>
      <plugins>
        <!--
        plugins
        * maven-source-plugin
        * maven-javadoc-plugin
        * maven-gpg-plugin
        * nexus-staging-maven-plugin
        -->
      </plugins>
    </build>
  </profile>
</profiles>
```

➔ settings.xml 설정

```
<profiles>
  <profile>
    <id>release</id>
    <activation>
      <activeByDefault>true</activeByDefault>
    </activation>
    <properties>
      <gpg.executable>gpg</gpg.executable>
      <gpg.passphrase>${env.MAVEN_GPG_PASSPHRASE}</gpg.passphrase>
    </properties>
  </profile>
</profiles>
```

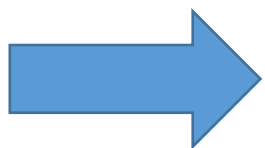
A large, irregular blue ink splatter or blotch serves as the background for the text. The splatter has a textured, painterly appearance with various shades of blue and some white highlights, giving it a dynamic and artistic feel. It is centered on a plain white background.

CI(Continuous Integration) 적용

Travis CI 적용 사례

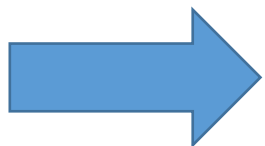
Considerations when applying CI

- GPG 적용을 위해서는 Secure Key 파일 형상 등록 필요
 - ➔ 파일 암호화 후 형상서버 등록
- OSSRH deploy를 위한 인증 정보 필요
 - ➔ 인증 정보 암호화 후 형상서버 등록 (settings.xml)



CI 환경에 따른 파일 및 데이터 암호화 적용

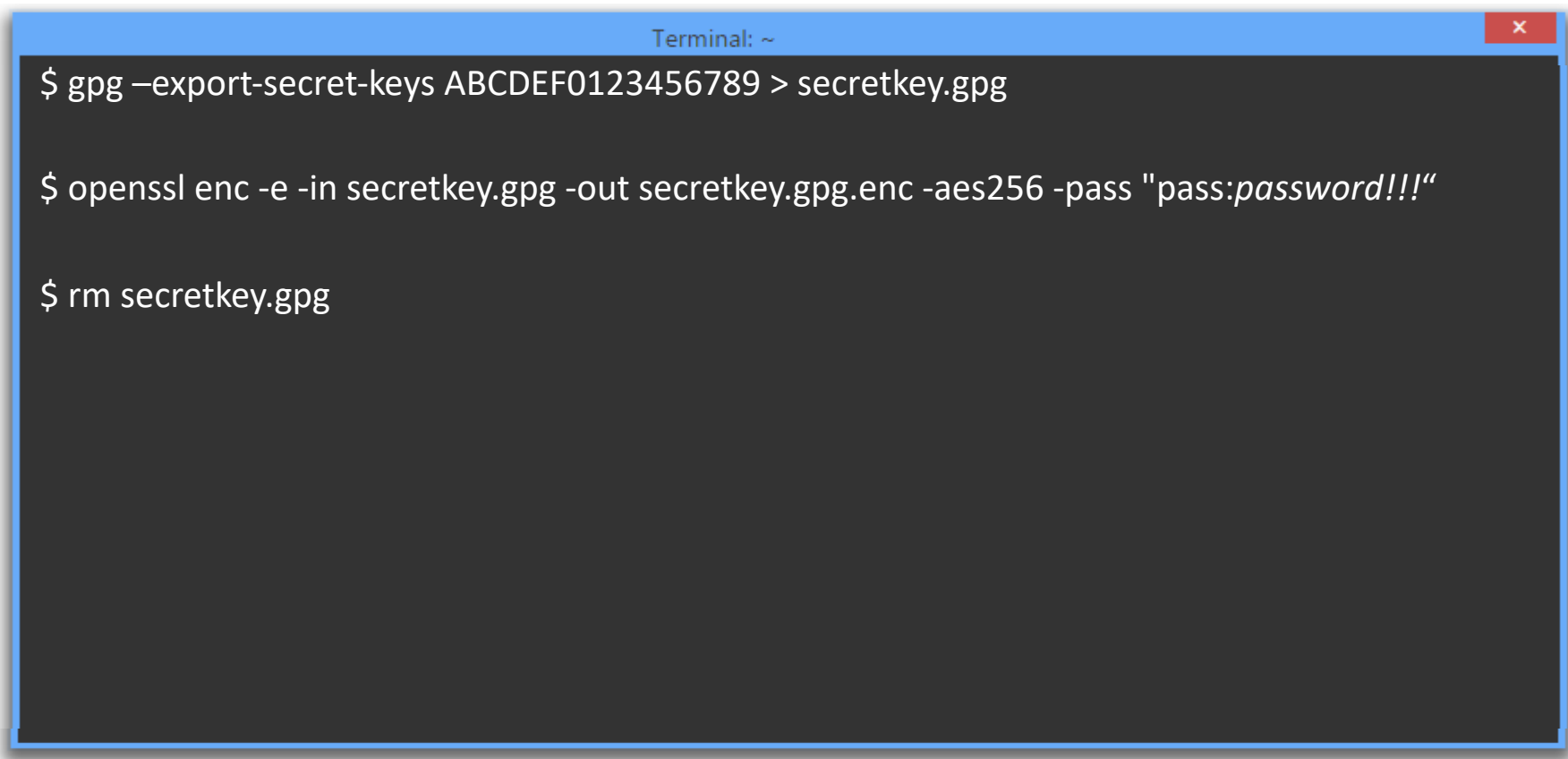
- Maven Profile 분리 실행 필요



조건별 릴리즈(Conditional release) 적용

Encrypting Files & Data

- File Encryption (AES-256 w/ openssl)

A terminal window with a blue title bar labeled "Terminal: ~" and a red close button. The terminal has a dark background with white text. It shows three commands being executed: 1. "\$ gpg --export-secret-keys ABCDEF0123456789 > secretkey.gpg", 2. "\$ openssl enc -e -in secretkey.gpg -out secretkey.gpg.enc -aes256 -pass 'pass:password!!!'", and 3. "\$ rm secretkey.gpg".

```
Terminal: ~  
$ gpg --export-secret-keys ABCDEF0123456789 > secretkey.gpg  
  
$ openssl enc -e -in secretkey.gpg -out secretkey.gpg.enc -aes256 -pass "pass:password!!!"  
  
$ rm secretkey.gpg
```

➔ 형상서버에
암호화된
secretkey.pgp.enc
파일 등록

Tip) openssl 1.0.X 버전으로 암호화해야 함 (Travis CI 현재 1.0.2g 버전 적용 중)

Encrypting Files & Data – cont'd

- File Decryption

- "secretkey.gpg.enc" 파일을 복호화하는 스크립트 구성 및 적용 (eg: setup_deploy.sh)

```
▶ setup_deploy.sh ×  
#!/bin/sh  
openssl enc -d -in secretkey.gpg.enc -out secretkey.gpg -aes256 -pass "pass:$SECRET_KEY_DEC_KEY"
```

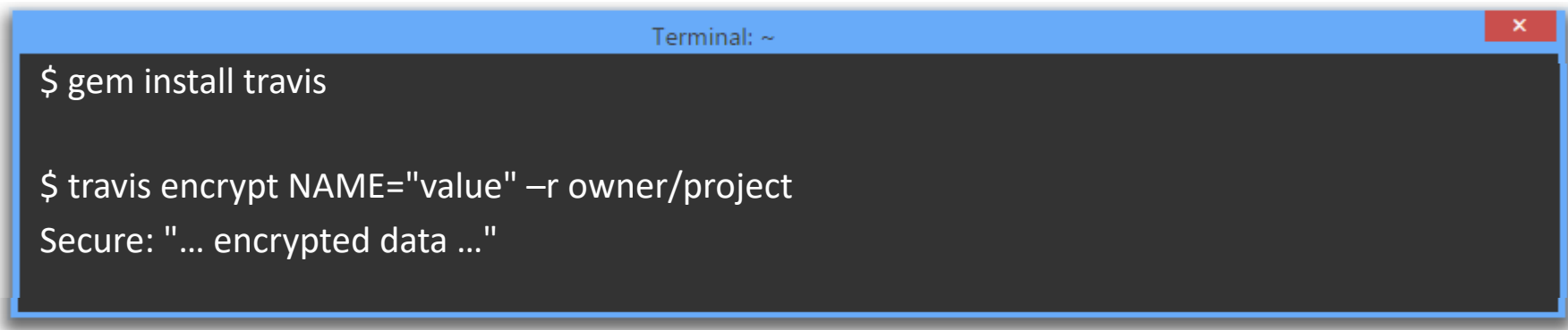
- Travis CI 설정 파일 적용 (.travis.yml)

```
YML .travis.yml ×  
before_install:  
- chmod +x mvnw  
- chmod +x setup_deploy.sh  
- ./setup_deploy.sh  
- gpg --import secretkey.gpg
```

Tip) before_install step은 기본 인프라 관련 패키지 설치 등에 사용 (추가 language 설치 등)

Encrypting Files & Data – cont'd

- Data Encryption (RSA w/ Travis CLI)

A terminal window with a blue title bar labeled "Terminal: ~". The terminal has a black background with white text. It shows the command "\$ gem install travis" followed by "\$ travis encrypt NAME='value' -r owner/project". The output is "Secure: '... encrypted data ...'".

```
Terminal: ~  
$ gem install travis  
  
$ travis encrypt NAME="value" -r owner/project  
Secure: "... encrypted data ..."
```

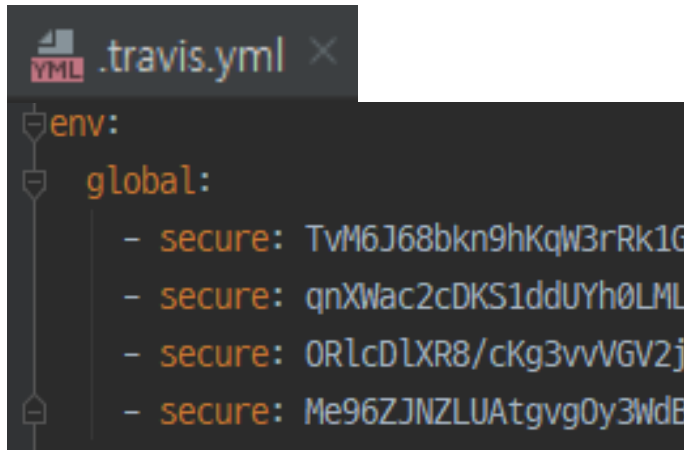
- 암호화 대상 (4개)
 - OSSRH user / password
 - GPG 보안키(securekey) passphrase
 - GPG 보안키 복호화 키 (AES 256 키)

또는 <http://rkh.github.io/travis-encrypt/public/>
에서도 암호화 가능
(JavaScript를 통한 암호화 지원)


Encrypting Files & Data – cont'd

- Data Decryption

- Travis CI 설정 파일 적용 (.travis.yml)



```
.travis.yml
env:
  global:
    - secure: TvM6J68bkn9hKqW3rRk1G
    - secure: qnXWac2cDKS1ddUYh0LML
    - secure: 0RlcDLXR8/cKg3vvVGV2j
    - secure: Me96ZJNZLUAAtgvg0y3WdE
```

- 
- MAVEN_REPO_USERNAME (Maven settings.xml)
 - MAVEN_REPO_PASSWORD (")
 - MAVEN_GPG_PASSPHRASE (")
 - SECRET_KEY_DEC_KEY (setup_deploy.sh)

Encrypting Files & Data – cont'd

- Data Decryption – cont'd

- Maven settings.xml 파일

- OSSRH server 정보
- Maven gpg plugin 관련 설정

➔ 형상서버에 settings.xml
파일 등록

- setup_deploy.sh (before_install
스크립트



```
<servers>
  <server>
    <id>ossrh</id>
    <username>${env.MAVEN_REPO_USERNAME}</username>
    <password>${env.MAVEN_REPO_PASSWORD}</password>
  </server>
</servers>

<profiles>
  <profile>
    <id>release</id>
    <activation>
      <activeByDefault>true</activeByDefault>
    </activation>
    <properties>
      <gpg.executable>gpg</gpg.executable>
      <gpg.passphrase>${env.MAVEN_GPG_PASSPHRASE}</gpg.passphrase>
    </properties>
  </profile>
</profiles>
```



```
#!/bin/sh
openssl enc -d -in secretkey.gpg.enc -out secretkey.gpg -aes256 -pass "pass:$SECRET_KEY_DEC_KEY"
```

Conditional release

- Travis Job Lifecycle

- 0. VM 생성 / Repository clone
- 1. (optional) "apt addons" (참고 : OS Ubuntu 16.04.6 LTS, Docker 18.06)
- 2. (optional) "cache components"
- 3. "before_install"
- 4. **"install" : install any dependencies required (eg: mvn install)**
- 5. "before_script"
- 6. **"script" : run the build script (eg: mvn test)**
- 7. (optional) "before_cache"
- 8. "after_success" or "after_failure"
- 9. (optional) "before_deploy"
- 10. **(optional) "deploy"**
- 11. (optional) "after_deploy"
- 12. "after_script"

Conditional release

- Travis 설정 (.travis.yml)

- ② "release" profile을 적용하여
maven deploy 실행 ←
- ① master 빌드 시에 ←

```
# .travis.yml
language: java
jdk:
  - openjdk8
env:
  global:
    - secure: TvM6J68bkn9hKqW3rRk1GoTFt6w/02iZteBVS8XnM+Jff1YPZ1nF2Df8GK/m1eJ3o8EtqYh1
    - secure: qnXWac2cDKS1ddUYh0LMLDdpMZJBVMWQsr/V1JpvRBK1STFkQ3K0000FBGMAHFV+pqP3zFk5
    - secure: ORlcDlXR8/cKg3vvVGv2j/QdI7e7i0q0jnfxyzBcWIMn7drL0faWE2mNZFoAxMAode/7/pKZy
    - secure: Me96ZJNZLUAtgvg0y3WdBS70cyAcQeL/s7hYfqUUsG9WwPzoXC96gGGGuA4EQgiR/fSXXQRy
  before_install:
    - chmod +x mvnw
    - chmod +x setup_deploy.sh
    - ./setup_deploy.sh
    - gpg --import secretkey.gpg
  deploy:
    provider: script
    script: "._/mvnw clean deploy -DskipTests=true -P release --settings settings.xml"
    skip_cleanup: true
  on:
    branch: master
```

Build examples

RedCA-Family / code-analyst

build passing

Current

Branches

Build History

Pull Requests

Build #133

More options

✓ development CRON Merge pull request #13 from switchover/patch-duplication-app

#133 passed

Restart build

DuplicationApp csv parsing error fixed

Ran for 3 min 21 sec

7 days ago

Commit 09223f2

Branch development

Vincent Han authored GitHub committed

JDK: openjdk8 Java

Job log

View config

Remove log

Raw log

1 Worker information

6

7 Build system information

161

162

163 Installing openjdk8

164

165 \$ git clone --depth=50 --branch=development https://github.com/RedCA-Family/code-analyst.git RedCA-Family/code-analyst

169

170

worker_info

system_info

docker_mtu

resolvconf

install_jdk

git_checkout

0.11s

0.01s

3.09s

0.00s

4.42s

0.01s

Reference Project

The screenshot shows the GitHub interface for the repository `switchover / code-analysis`. The repository has 1 commit, 1 branch, 1 release, 1 contributor, and is licensed under Apache-2.0. The latest commit is `e4c4918` from 1 hour ago. The repository contains the following files and folders:

File/Folder	Commit	Time
<code>.mvn/wrapper</code>	first commit	16 hours ago
<code>src</code>	first commit	16 hours ago
<code>.editorconfig</code>	first commit	16 hours ago
<code>.gitignore</code>	first commit	16 hours ago

The repository page includes a search bar, navigation links (Pull requests, Issues, Marketplace, Explore), and a sidebar with links to Code, Issues, Pull requests, Projects, Wiki, Security, Insights, and Settings. The repository name is `switchover / code-analysis`, and the license is Apache-2.0. The repository has 1 commit, 1 branch, 1 release, 1 contributor, and is licensed under Apache-2.0. The latest commit is `e4c4918` from 1 hour ago. The repository contains the following files and folders:



고맙습니다.

Q&A

References

- Maven Central Repository
 - <https://maven.apache.org/repository/index.html>
- Guide to uploading artifacts to the Central Repository
 - <https://maven.apache.org/repository/guide-central-repository-upload.html>
- OSSRH Guide
 - <https://central.sonatype.org/pages/ossrh-guide.html>
- Encryption and Digital Signatures using GPG
 - <https://cran.r-project.org/web/packages/gpg/vignettes/intro.html>
- Travis CI Guide
 - <https://docs.travis-ci.com/>

[참조] Gradle 적용 시

- Metadata & signing (build.gradle)
 - OSSRH 적용시 pom.xml 파일 필요
 - Javadoc 및 source jar 생성
 - GPG 전자서명

```
signing.keyId=KeyId  
signing.password=Password  
signing.secretKeyRingFile=PathToGPGFile  
  
ossrhUsername=jira-username  
ossrhPassword=jira-password
```

→ gradle.properties 설정

```
apply plugin: 'maven'  
apply plugin: 'signing'  
  
group = "com.samsungsds.analyst"  
archivesBaseName = "code-analyst"  
version = "2.9.0"  
  
task javadocJar(type: Jar) {  
    classifier = 'javadoc'  
    from javadoc  
}  
  
task sourcesJar(type: Jar) {  
    classifier = 'sources'  
    from sourceSets.main.allSource  
}  
  
artifacts {  
    archives javadocJar, sourcesJar  
}
```

[참조] Gradle 적용 시 – cont'd

- uploadArchives 설정

```
uploadArchives {
    repositories {
        mavenDeployer {
            beforeDeployment { MavenDeployment deployment -> signing.signPom(deployment) }

            repository(url: "https://oss.sonatype.org/service/local/staging/deploy/maven2/") {
                authentication(userName: ossrhUsername, password: ossrhPassword)
            }

            snapshotRepository(url: "https://oss.sonatype.org/content/repositories/snapshots/") {
                authentication(userName: ossrhUsername, password: ossrhPassword)
            }
        }

        pom.project {
            name 'Code-Analyst'
            packaging 'jar'
            // optionally artifactId can be defined here
            description 'Code Analyst to measure various code metrics'
            url 'https://github.com/RedCA-Family/code-analyst'
```

[참조] Gradle 적용 시 – cont'd

- uploadArchives 설정 – cont'd

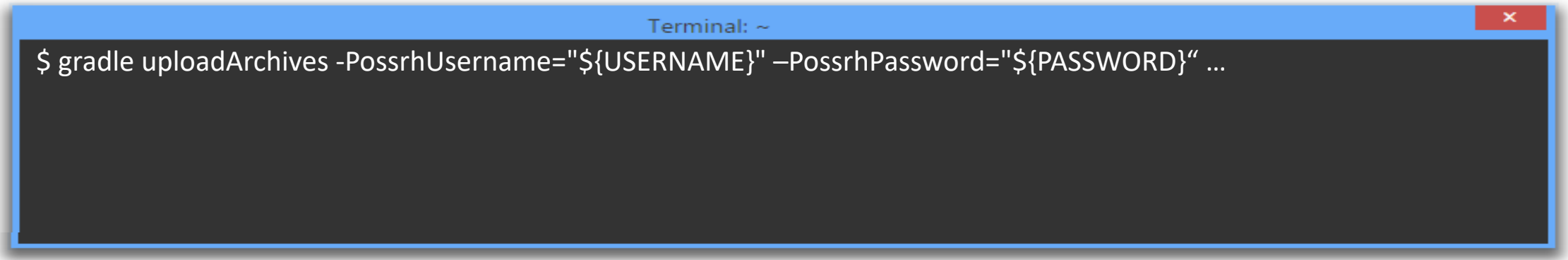
```
scm {
    connection 'scm:git:https://github.com/RedCA-Family/code-analyst.git'
    developerConnection 'scm:git:https://switchover@github.com/RedCA-Family/code-analyst.git'
    url 'https://github.com/RedCA-Family/code-analyst'
}

licenses {
    license {
        name 'The Apache License, Version 2.0'
        url 'http://www.apache.org/licenses/LICENSE-2.0.txt'
    }
}

developers {
    developer {
        name 'Vincent Han'
        email 'switchover@gmail.com'
    }
}
```

[참조] Gradle 적용 시 – cont'd

- uploadArchives 설정 – cont'd


A terminal window with a blue title bar and a dark gray body. The title bar contains the text "Terminal: ~" and a red close button. The body contains a single line of text: "\$ gradle uploadArchives -PossrhUsername="\${USERNAME}" -PossrhPassword="\${PASSWORD}" ...".

```
Terminal: ~  
$ gradle uploadArchives -PossrhUsername="${USERNAME}" -PossrhPassword="${PASSWORD}" ...
```

[참조] Jenkins 암호화 처리

- Credential Plugin 사용 – secret text

[Back to credential domains](#)

 **Add Credentials**

Kind

Secret text ▼

Secret

ID

?

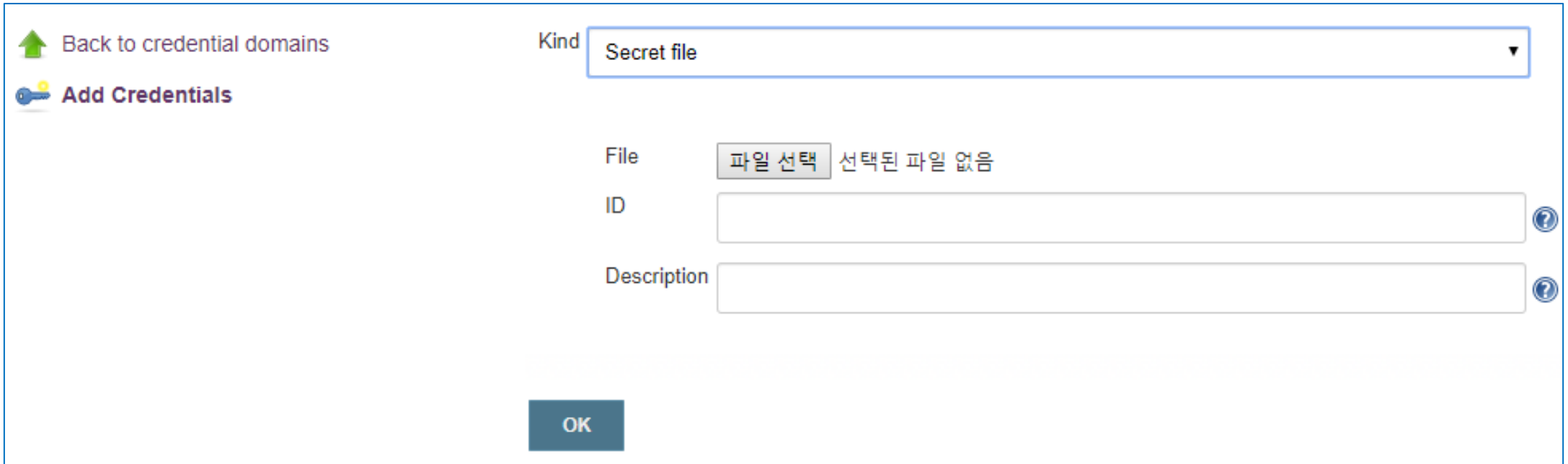
Description

?

OK

[참조] Jenkins 암호화 처리 – cont'd

- Credential Plugin 사용 – secret file



The screenshot shows the 'Add Credentials' dialog in Jenkins. On the left, there are two links: 'Back to credential domains' with a green arrow icon and 'Add Credentials' with a key icon. The main area is titled 'Kind' and has a dropdown menu set to 'Secret file'. Below this, there are three fields: 'File' with a button labeled '파일 선택' (File Select) and the text '선택된 파일 없음' (No file selected); 'ID' with an empty text box and a help icon; and 'Description' with an empty text box and a help icon. At the bottom center is an 'OK' button.

Back to credential domains

Add Credentials

Kind: Secret file

File: 선택된 파일 없음

ID:

Description:

OK

[참조] Jenkins 암호화 처리 – cont'd

- <사용> Credential Binding Plugin 사용

