artifact관리툴 - nexus 구축하기

시작하는 말

안녕하세요, 고니입니다.

기존에 작성했던 컨텐츠들 업데이트를 하면서 문서의 리팩토링(Refactoring)을 진행해보려고 합니다.

Devop pipeline의 산출물(artifact)을 관리할 수 있는 대표적인 툴이 Sonatype에서 제공하는 Nexus와 JFrog에서 제공하는 jfrog 인데요.

두가지 툴의 장단점 비교는 별도로 작성해 볼 예정이고, 가장많이 사용하고 있는 Nexus를 소개해보려고 합니다.

Nexus 소개

1. NeXus Repository Manager (약어;NXRM)
2. 지원 OS : Windows / Linux / Unix / OSX
3. 설치버전 : 3.29.2 OSS ('21.01.10 기준)
4. License : Eclipse Public License-v 1.0
5. Pro버전 추가 기능 : Atalssian Crowd지원, Staging / Build 분리 / 태깅 / 계정별 토큰 / HA / 아티팩트별 OSS보안 취약점 진단 / 기술지원 / 그룹 저장소 관리 기능 / SAML / SSO 기능

Nexus 시스템 요구사항

1. hardware requirements

Type	**Minimal**	**requirement**
CPU	4Core	8Core
Memory	Host : 8Gb JAVA Heap : 2Gb	Host : 8Gb Java Heap : 4GB
Disk	330M 이상	330M 이상

- 사용가능 디스크 공간이 4GB이하일 경우 Readonly모드로 작동
- Java Directory 메모리 설정공식은 (호스트메모리 * 0.6) - java heap

2. 지원 가능 파일시스템

파일시스템 종류	**Embbed**	**Blob**	비 고
XFS	O	O	
NFS V4	O	O	
NFS V3	X	X	오류발생
AWS - EBS	O	O	
AWS - EFS	X	O	Embbed 응답 이슈
AWS - S3	X	O	임베디드 데이터 적용불가
AWS- S3호환	X	X	S3 호환 저장소는 지원 불가
Samba/CIFS	X	O	임베디드 데이터 적용불가
GlusterFS	X	X	Split-brain 발생 , 응답속도 지연
Fuse	X	X	데이터 오류 발생

* Embbed : 응답성이 빠른 스토리지 (DB / Elastic Search)

* Blob : binary 저장소

3. 디렉토리 구조

디렉토리명	설 명
bin	시작스크립트 / 시작관련 구성파일
etc	nxrm 구성파일 /런타임 구성파일
lib	nxrm 라이브러리 파일
public	nxrm 공개 리소스데이터
system	프로그램 구성 데이터 / 플러그인
blobs	blob 저장소
cache	캐시된 데이터
db	orientDB 데이터 저장
elasticsearch	현재 구성된 elasticsearch 데이터
health-check	저장소 상태 확인 기능 보고서
keystores	저장소 관리자를 식별하는 키 데이터
log	로그데이터 nexus.log → 관리자 프로그램 로그 , request.log → http요청로그 jvm.log → jvm stdout, stderror 및 덤프데이터 karaf.log → apache karaf 컨테이너 로그 파일 audit → 감사 활성시 감사로그 데이터 staks → 작업별 수행한 로그 tmp → 임시저장소

Nexus 기능

1. 리포지터리 종류

- hosted : 서버에 저장되는 리포지터리
- proxy : hosted와 연결하여 사용하는 cache용 리포지터리

- group : 여러개의 hosted or proxy를 그룹으로 묶은 논리개념의 리포지터리

2. 리포지터리 버전관리 방식

- release : 정식릴리즈되는 아티팩트 저장소 / 동일버전으로 한번만 업로드 가능 (필요시 삭제 후 업로드해야 함)
- snapshot : 수시로 릴리즈 되는 아티팩트 저장소 / 같은 버전으로 여러번 업로드 가능

3. 리포지터리 유형

- maven2 : 메이븐에서 제공하는 메타데이터로 버전관리 (group / artifact id / version의 메타데이터로 관리함)
- raw : 별도의 메타데이터는 없으며 파일 위치와 파일이름만 생성
- yum : yum 리포지터리 저장소

4. 아티팩트 관리 방식

- curl을 이용한 바이너리 관리
- test_repo에 있는 img-1.png 파일 다운로드

```
$ curl -u 'test:test123' http://192.168.10.10:11000/repository/test_repo/img-1.png -o img-1.png
```

- test.tar.gz 파일을 test_repo로 업로드

```
$ curl -u 'test:test123' --upload-file
./test.tar.gz http://192.168.10.10:11000/repository/test_repo/test.tar.gz
```

- 다중파일 업로드시

```
$find ./ -exec curl -u 'test:test123' -T
./{} http://192.168.10.10:11000/repository/test_repo/test.tar.gz/{} \;
```

- test_repo에 있는 test.tar.gz 파일 삭제

```
$> curl --request DELETE -u
'test:test123' http://http/192.168.10.10:11000/repository/test_repo/test.tar.gz
```

Nexus 설치 (Docker 기반)

1. container pull 하기

```
$> docker pull sonatype/nexus3:3.29.2    #2.x을 사용할 경우 sonatype/nexus를 사용하면 됨
```

2. nexus 저장용 데이터 생성하기

```
$> mkdir /data/nexus-repo
$> chown -R 200 /data/nexus-repo/
```

3. container 실행하기

```
$ docker run -d -p 88081:8081 \
-e TZ=Asia/Seoul  \
-v /etc/localtime:/etc/localtime:ro \
-v /data/nexus-repo/:/nexus-data \
-name nexus-repo sonatype/nexus3:3.29.2
```

4. Container log 확인 (started sonatype 메시지가 출력되면 정상구동 완료)

```
$ docker logs -f nexus-repo
...
2021-01-19 23:32:02,793+0900 INFO  [jetty-main-
1] *SYSTEM org.sonatype.nexus.repository.httpbridge.internal.ViewServlet - Initialized
2021-01-19 23:32:02,870+0900 INFO  [jetty-main-
1] *SYSTEM org.eclipse.jetty.server.handler.ContextHandler -
 Started o.e.j.w.WebAppContext@637c45d9{Sonatype Nexus,/,file:~/~//opt/sonatype/nexus/public/,A
VAILABLE}
2021-01-19 23:32:02,928+0900 INFO  [jetty-main-
1] *SYSTEM org.eclipse.jetty.server.AbstractConnector -
 Started ServerConnector@45bb642d{HTTP/1.1, (http/1.1)}{0.0.0.0:8081}
2021-01-19 23:32:02,929+0900 INFO  [jetty-main-1] *SYSTEM org.eclipse.jetty.server.Server -
 Started @41677ms
2021-01-19 23:32:02,930+0900 INFO  [jetty-main-
1] *SYSTEM org.sonatype.nexus.bootstrap.jetty.JettyServer -
~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-
~-~-~-~-~--
\\Started Sonatype Nexus OSS 3.29.2-02
\\~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-~-
~-~-~-~-~-~--
```

5. web-ui 접속하기 – http://{{서버 IP}}:88081/

6. 초기 접속 화면

7. 로그인 패스워드 확인

```
$docker exec -it nexus cat /nexus-data/admin.password
1123-aad-xxxx-xxxx-
```

8. 로그인 계정 (admin)

9. admin 패스워드 설정 (변경할 패스워드 입력 후 Next)

10. 마침

Repository Cleanup 정책관리

1. cleanup policies 정책 수립하기
    - 상단 톱니바퀴 모양 → Repository → Cleanup Policies → Create Cleanup Policy
    - Cleanup policy 규칙 메뉴 구성
    - 항목별 세부구성 (예시는 90일 보관 후 삭제하는 정책)

항 목	설 명	설정값	비 고
Name	규칙이름 설정	proxy_cleanup_{{보관일}}d	고유의 값 설정
Format	Repository format 설정	raw	모든 유형의 repository로 할 경우 all formats 선택
notes	규칙에 대한 설명정보		옵션
Component Age	저장된 데이터의 보관 주기	90	체크박스 선택 후 보관일 입력
Component Usage	다운로드 되지 않은 데이터 보관주기	90	체크박스 선택 후 보관일 입력
preview repository	해당 정책을 수립시 삭제되는 데이터 리스트가 보임		

Repository 관리

1. Hosted 기반의 리포지터리 생성

- 상단 톱니바퀴 모양 → Repository → Create Repository → RAW (hosted)
- repository 생성 메뉴구성
- 항목별 세부설명

항 목	설 명	설정값	비 고
Name	Repository 고유 이름	repo 이름 입력	문자, 숫자로 구성되어야 하고 특수문자는 (-, _)만 허용되고 마침표로 마칠수 없음
Online	repository 활성화 여부	체크	기본값은 체크되어 있음
Content Disposition	컨텐츠 관리 방법	attachment	Attachment / inline 중 택 1, Attachment로 구성해야 파일 전송이 유용함
Blob store	바이너리 저장 경로	default	별도 바이너리 저장소를 설정하지 않는 경우 default로 구성

항목	설명	설정값	비고
Deployment Policy	아티팩트 배포 / 업데이트 허용 여부	Allow redeploy	Allow redeploy : 재 배포허용 Deny redeploy : 재배포 불가 read only : 업데이트 불가 , 읽기전용으로 구성
Cleanup policies	Repository 오래된 파일 삭제		Cleanup 정책 수립했을때 적용

2. Proxy 기반의 리포지터리 생성

- 상단 톱니바퀴 모양 → Repository → Create Repository → RAW (Proxy)
- repository 생성 메뉴구성
- 항목별 세부설명

항목	설명	설정값	비고
Name	Repository 고유 이름		문자 , 숫자로 구성되어야 하고 특수문자는 (-, _)만 허용되고 마침표로 마칠수 없음
Online	repository 활성화 여부		기본값음 체크되어 있음
Remote storage	hosted 리포지터리 주소 입력		
Blocked	하위 Proxy 연결을 차단할 경우 체크	체크수행	
Auto blocking enabled	Remote Storage에 연결이 되지 않을 경우 하위 Proxy 연결 차단	체크수행	
Maximum component age	cache된 바이너리의 최대 보관 주기 (분단위)	60	

항목	설명	설정값	비고
Maximum metadata age	cache된 메타데이터 최대 보관 주기 (분단위)	60	
Blob store	바이너리 저장 경로	default	별도 Storage로 구성할 경우 구성한 Stoage 이름 선택
Strict Content Type Validation	바이너리 저장시 MIME유형여부 확인	체크 해제	
Not found cache enabled	원격 저장소에 없는 컨텐츠 요청시 캐쉬로 응답	체크 수행	
Not found cache TTL	원격 저장소에 파일없다는 캐쉬 정보 보관주기 (분단위)	60	
Cleanup policies	Repository 오래된 파일 삭제		Cleanup 정책 수립했을때 적용
HTTP authentication	원격저장소의 계정명 입력	host용 nexus에 등록한 계정정보 입력	

3. Proxy Repo에 cache 정보 강제 삭제
   - 톱니바퀴 → Repository → Proxy타입의 Repository 선택 → Invalidate cache 선택

Proxy 기능 테스트

1. 보관 주기별 Proxy 데이터 전송

보관주기	**Proxy** 반영시간
-1	수동으로 캐쉬 삭제전까지 반영 안됨
0	즉시
1	1분뒤 Proxy repo에 반영
2	2분뒤 Proxy Repo에 반영

2. Proxy 전달 Traffic 정보

- 테스트정보 : Proxy에 1차버전 caching 된 상태에서 host에 새로운 바이너리가 업데이트 된 경우, caching flush 후 다시 caching하는 과정 확인 (flush되기 전까지 지속적 호출)
- 패킷 정보 (Hosted → Proxy), 192.168.10.10 (Proxy), 192.168.100.10 (Hosted)

```
15:42:31.554851 IP 192.168.10.10.42224 > 192.168.100.10.88081: Flags [S], seq 1521618240, win 29200, options [mss 1460,nop,nop,sackOK], length 0
15:42:31.555029 IP 192.168.100.10.88081 > 192.168.10.10.42224: Flags [S.], seq 2385004980, ack 1521618241, win 29200, options [mss 1460,nop,nop,sackOK], length 0
15:42:31.555223 IP 192.168.10.10.42224 > 192.168.100.10.88081: Flags [.], ack 1, win 29200, length 0
15:42:31.555234 IP 192.168.10.10.42224 > 192.168.100.10.88081: Flags [P.], seq 1:262, ack 1,
```

Revision #6

Created 7 June 2022 01:25:55 by artop0420

Updated 1 September 2024 00:19:05 by artop0420