**Configuration Management**

**ServerManager Guide for Git**

**스마트워크 2, 3단계 구축\_형상관리\_가이드**

****Version 1.0

2016.09.05

작성자 : 문해민



**Table of Contents**

[**1.** **개요** - 2 -](#_Toc460998713)

[**2.** **원격 저장소 [ 서버 ] 접속 – CI Server** - 2 -](#_Toc460998714)

[**2.1.** **Putty 설치** - 3 -](#_Toc460998715)

[**2.2.** **SSH 원격 접속** - 4 -](#_Toc460998716)

[**3.** **서버 관리하기** - 5 -](#_Toc460998717)

[**3.1. Git 사용자 계정 관리 [ ncp, ncp\_workplace ]** - 5 -](#_Toc460998718)

[**3.1.1.Git 사용자 생성 및 추가** - 5 -](#_Toc460998719)

[**3.1.2.Git 사용자 삭제** - 8 -](#_Toc460998720)

[**3.2.** **서버 사용자 관리** - 9 -](#_Toc460998721)

[**3.3.** **작업 이련 관리** - 9 -](#_Toc460998722)

1. **개요**

**Abstract**

본 문서는 형상관리 도구인 Git Server [CI Server]를 관리하기 위한 초기 설정 및 방법들을 제공합니다. Git Server에서는 다양한 툴과 설정 등을 통해 디버깅, 코드 인스펙션, 로그를 통한 충돌 감지 등 여러가지 작업들이 가능하게 됩니다. 하지만 본 문서에서는 **‘동시다발적인 협업, 인터넷 환경이나 장소에 제약 받지 않는 작업 환경 구축’**에 초점을 맞춰 **‘사용자 인증 및 관리에 대한 전반적인 가이드에 범위’**를 두고 있습니다.

1. **원격 저장소 [ 서버 ] 접속 – CI Server**

Git Server에 접속하기 위해선 SSH(Secure Shell)를 이용해야 합니다. Putty라는 SSH 접속 프로그램으로 서버에 원격 접속하도록 하겠습니다.

* 1. **Putty 설치**

**Putty Download URL**

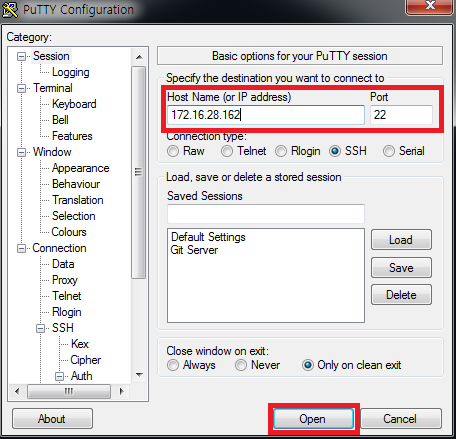
[**http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html**](http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html)

* 해당 URL을 접속하시면 다음과 같은 화면이 보여지게 됩니다.



* 위와 같은 화면에서 본인의 PC환경에 맞는 “putty.exe” 파일을 다운로드 합니다.
  1. **SSH 원격 접속**

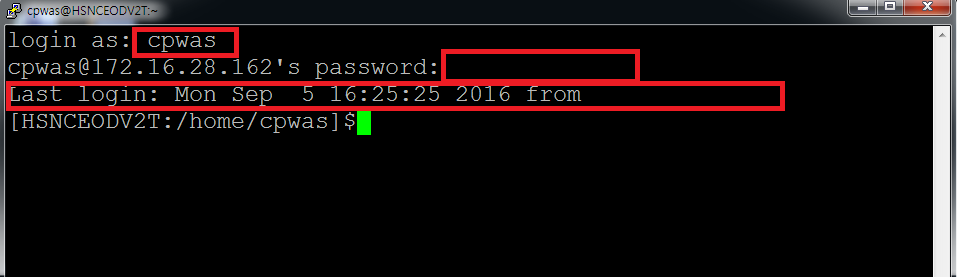
‘putty.exe’ 프로그램을 실행하시면 다음과 같은 화면이 보여집니다.



* 위와 같은 화면에서 Host Name의 Input box에 접속하고자 하는 IP 주소를 기입합니다.

***[S&C 협업포탈 원격 저장소의 경우 172.16.28.162 / SSH 접속 개방 포트 22]***

* IP 주소 기입 후 Open 버튼을 클릭하면 아래와 같은 화면이 보여집니다.



서버에는 사전에 원격 접속을 허용하기 위해 SSH Key 인증이 완료된 Admin 아이디가 존재하는데, 이를 통해 로그인 절차를 진행하시면 됩니다. 로그인이 완료되면 “Last login : 접속 날짜와 시각 from 본인의 IP” 문구를 확인 하실 수 있습니다.

***[ S&C 협업포탈 “Admin ID / password : cpwas / 0916\_cpwas ]***

1. **서버 관리하기**
   1. **Git 사용자 계정 관리 [ ncp, ncp\_workplace ]**

기본적으로 ‘S&C 협업포탈’ Git 서버에는 “ncp, ncp\_workplace” Repository가 존재합니다.

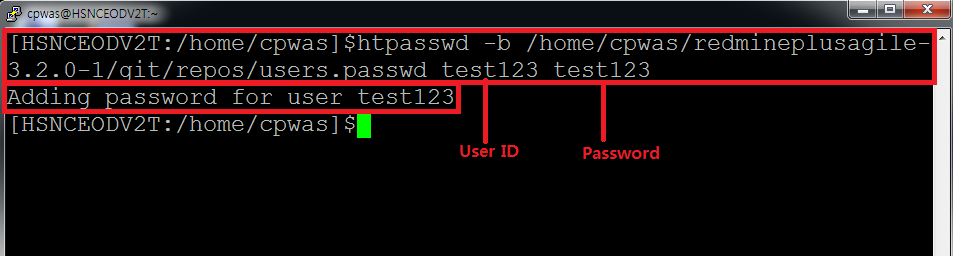
**ncp 프로젝트는 협업포탈 中 워크플레이스를 제외한 모든 기능**

**ncp\_workplace 프로젝트는 협업포탈 中 워크플레이스 기능**

두 가지 프로젝트가 하나의 협업포탈을 구축하게 되는 것 입니다. 서비스하는 것은 하나의 서비스지만 프로젝트가 두 개로 분리되어 있기 때문에 Git 사용자 계정을 부여할 시 두 개의 인증을 모두 해주어야 합니다.

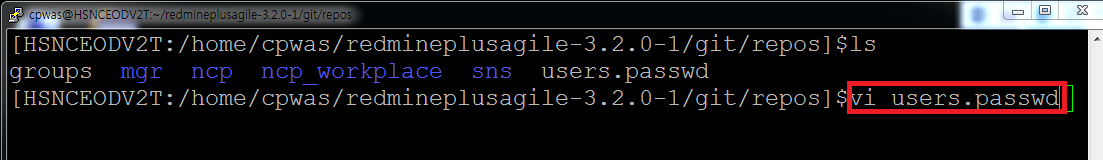
* + 1. **Git 사용자 생성 및 추가**
* 서버에 접속이 완료된 상태에서 다음과 같은 명령어를 입력합니다.

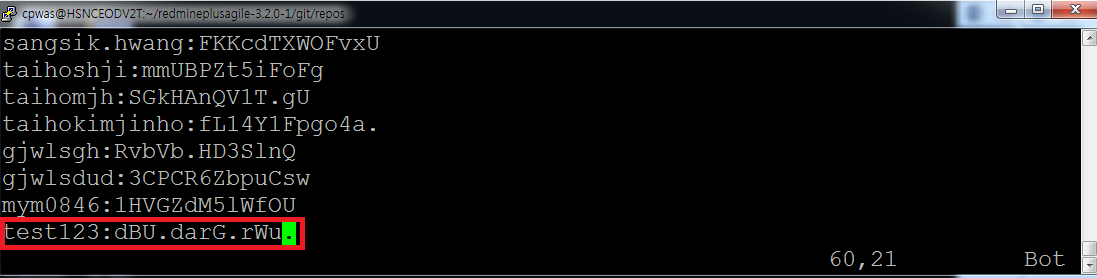
**htpasswd -b /home/cpwas/redmineplusagile-3.2.0-1/git/repos/users.passwd <user> <passwd>**



위 이미지에서는 **“test123/test123”** 이라는 ID/PW로 사용자 계정을 생성하고 있습니다. 계정이 성공적으로 생성되면 **“Adding password for user test123”** 라는 문구를 확인하실 수 있습니다.

계정은 ‘/home/cpwas/redmineplusagile-3.2.0-1/git/repos’ 라는 디렉토리의 ‘users.passwd’ 파일안에 암호화되어 추가되는데, 확인을 위해 cd 명령어를 이용하여 해당 디렉토리로 이동한 후 vi편집기를 이용하여 파일을 열어보도록 하겠습니다.





위 이미지와 같이 htpass 명령어를 사용해 생성한 test123 계정이 users.passwd 파일안에 암호화되어 추가된 것을 확인 하실 수 있습니다.

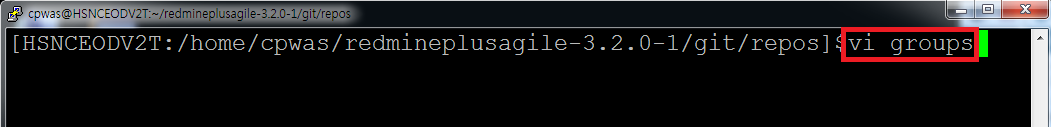
계정 생성은 완료되었으나, 아직 Repository group에 생성된 계정을 추가해주지 않았기 때문에 현재 상태에서는 사용자들이 자신의 워킹 디렉토리에 Repository를 복제해 올 수 없는 상태입니다.

Repository group에 생성한 계정을 추가하기 위해서는 다음과 같은 작업을 수행해야 합니다.

* ‘/home/cpwas/redmineplusagile-3.2.0-1/git/repos’ 디렉토리로 이동한 후, ls 명령어로 디렉토리를 조회하시면 ‘groups’ 라는 파일이 존재하는 것을 확인 하실 수 있습니다.

C:\Users\한화에스엔씨\Desktop\putty4.PNG

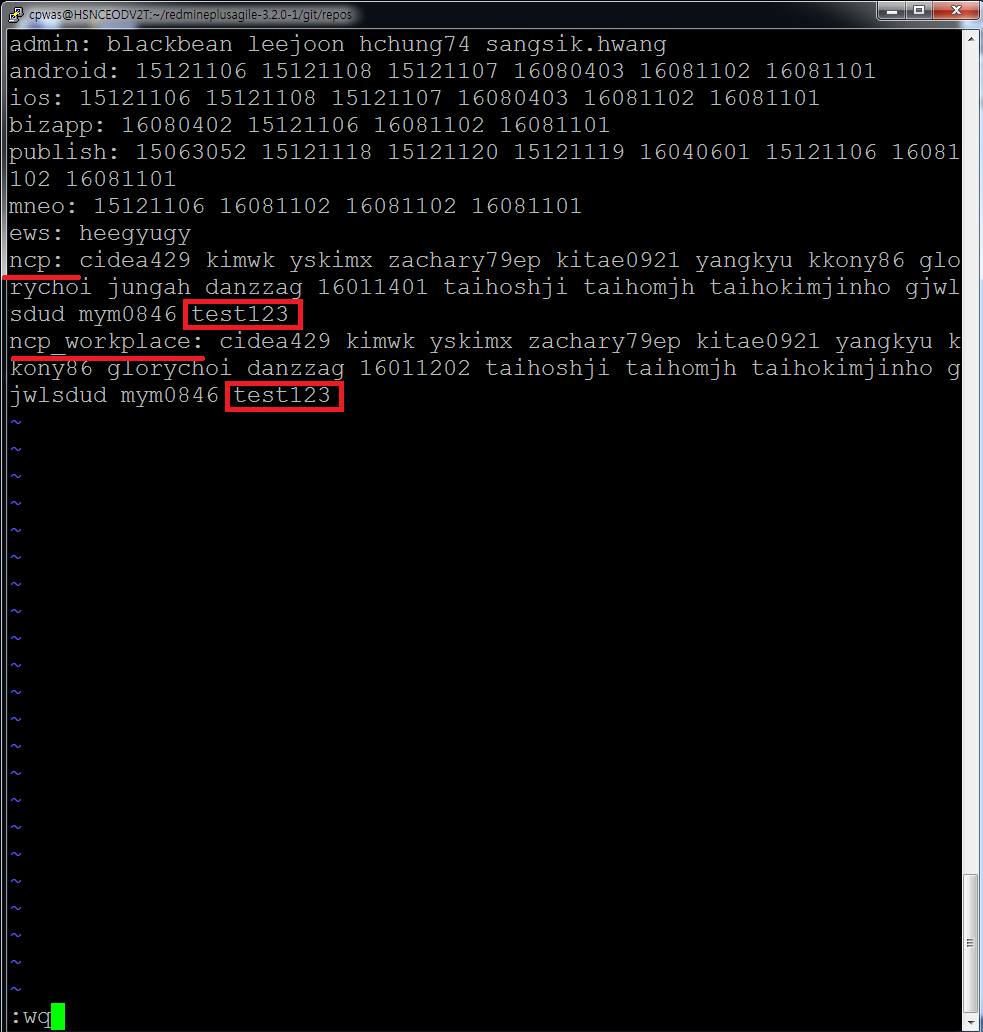
* Vi 편집기를 이용해서 groups 파일 편집을 준비합니다.



* ‘groups’ 파일이 오픈되면, 아래 그림과 같이 파일을 편집하시면 됩니다.

**ncp, ncp\_workplace 라인 모두 space(띄어쓰기) 구분자로 순서대로 기입해주시면 됩니다.**

ex) Group Name : userId1 userId2 userId3



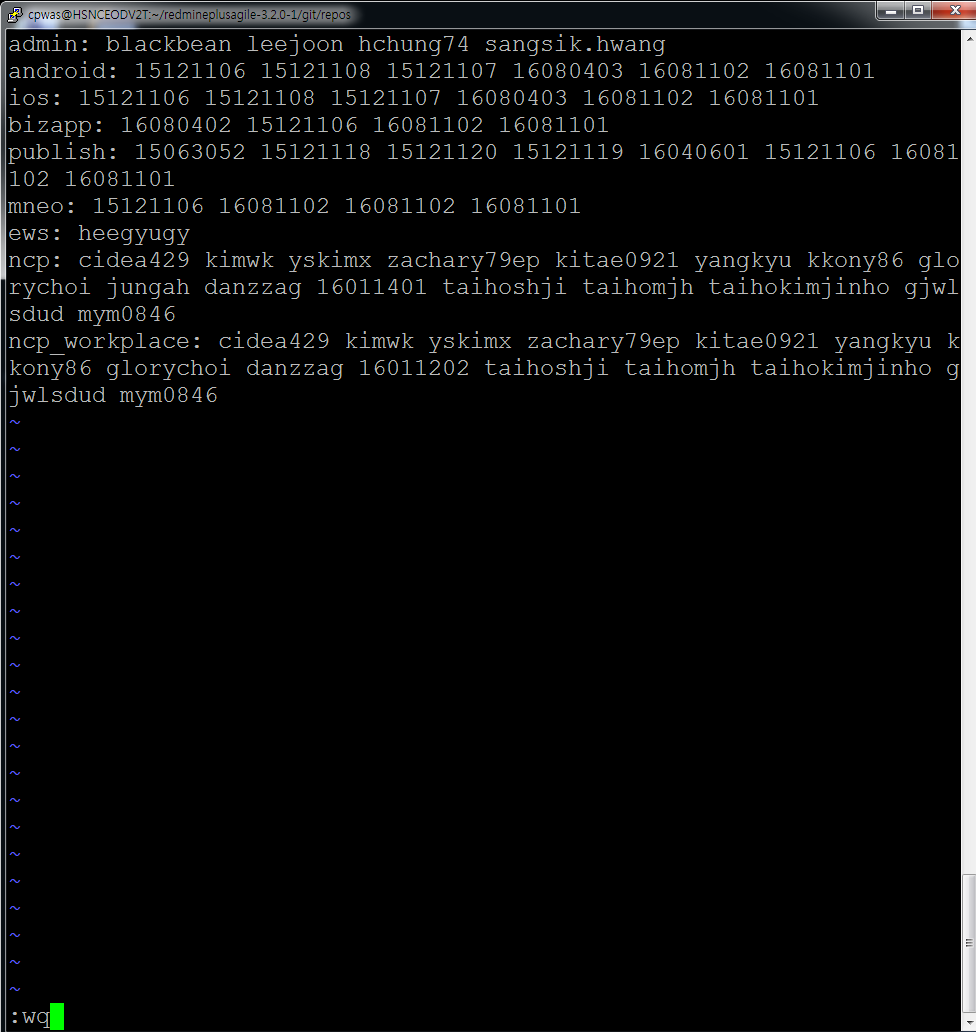
파일 편집을 완료하시고 vi편집기를 종료하시면 **‘Git 사용자 계정 생성 및 인증’** 절차가 모두 완료됩니다. 여기까지의 프로세스가 완료되면 사용자는 원격 저장소에 접근하여 협업할 수 있는 권한을 갖게 됩니다. **[Repository 복제, Push, Pull……]**

* + 1. **Git 사용자 삭제**

사용자 권한 회수는 vi 편집기를 이용해서 사전의 **‘계정 생성 및 인증 추가’** 작업했던 것을 그대로 지워준 후 저장해주시면 됩니다.

**vi users.passwd -> test123:암호화** 부분 삭제

**vi groups -> ncp, ncp\_workplace : ……. test123** 부분 삭제

****

* 1. **서버 사용자 관리**

Git 서버의 원격 저장소에 접근하는 것과 Git 서버에 원격 접속하는 것은 확연히 다른 개념입니다. Git 서버에 원격 접속한다는 것은 SSH Key 인증을 통해 허가한 사용자 아이디만 접근이 가능한 것입니다. 현재 S&C Git 서버에는 ‘cpwas’라는 공용 계정을 통해 운용하고 있습니다. 하지만 계정을 여러 개 생성하여 사용자를 분할하면 ‘chmod’ 권한 부여 명령어를 통해 서버에 접근하는 사용자들에게 읽기, 쓰기 권한을 제어할 수 있게 됩니다. 이는 서버에 대한 사용자 권한을 제어함으로써 서버 보안을 강화시킬 수 있게 됩니다.

* 1. **작업 이련 관리**

관리자는 서버에 대한 관리능력도 필요로 하지만 서버의 핵심인 ‘Git Repository’에 대한 관리 지식도 필요합니다.

Commit, Branch, Merge 등의 작업들이 git 사용자들로부터 동시적으로 발생하게 되는데 이때 여러가지 충돌상황들이 발생할 수 있습니다.

**git log**

**git branch**

**git status**

기본적으로 충돌이 발생하면 위와 같은 로그 조회 명령어를 통해 문제를 인식하고 발단 부분을 검출하여 문제해결을 신속하게 할 수 있어야 합니다.

**Git Reference URL**

<http://backlogtool.com/git-guide/kr/reference/>

<https://git-scm.com/doc>