Track Configurations using Git

Anas Mughal
anas@devsenesce.com
Topics

● Background and Overview
  - Types of configurations
  - Benefits of tracking configurations
  - Why DVCS

● Git Overview

● Configuration Tracking Steps

● Usage Examples

● DEMO

● Suggested Tools

● Conclusion
Background

- Software Development (20 years)
  - Empower and Enable Developers
- Setup Development Environments (5 years)
  - Open Source/Java/Android
  - Currently, Support Tooling for 150+ developers
- IT not managing our Tools servers
Software Development Tools

- Agile Planning
- Issue/Bug Tracking
- Dependency Management Server
- Continuous Build
- Continuous Code Analysis
- Source Code Control
- Developers (Virtual Desktops)
Configurations Overview

● Configurations are files that define
  - System/Server Settings
  - Application Settings

● Configurations are modified based on
  - Environments
  - Business Needs
Types of Configurations

- System Configurations
  - OS Settings/Services
  - Authentication/Authorization

- Software Configurations
  - Applications/Servers

- Application Integration Configurations

- IDEs
Configuration Examples

● System Configurations
  - Folder “etc”
  - Folder “init.d”
  - Files: hosts, sudoers, etc.

● Software Configurations
  - Apache Web Server Configuration
  - Application Server (e.g. Tomcat, Websphere)
  - Database Servers

● Entire IDE setups – plug-ins, updates, etc
Typical Approaches

- Back up configurations
  - same folder, or
  - backup folder
- Append date, etc.
- Make backup folders with comments
Track Configurations

using

Distributed Version Control System
Why use DVCS – Git/Hg

- DVCS are suited to track configurations
- Initialize a repository anywhere
- No central server
- No network needed
- Track changes to folders and/or files
- Ignore un-interesting folders/files
Benefits of Tracking

- Track changes – specially junior admins
- Undo and/or restore an earlier state
- Keep history of changes
- Backup entire history
  - push to a central/share location
- Restore/Revert
  - anytime on any workstation/server
Track Configurations with GIT
Git Overview

- Distributed Version Control System (DVCS)
- No Central Server
- Offline -- No Network
- FAST
- Efficient Branching and Merging
- Undo everything – and then Redo!
Git Limitations

● Performance Degrades for Large Repositories
● Limited Enterprise Support
  - Authentication/Authorization
  - Repository Management
  - SCM functions

Git Concepts

- Entire repository lives in “.git” folder
  - Trees, Blobs, Commits, Tags
- Track file system snapshots
  - Files = Blobs
  - Folders = Trees
- Ignore files/folders
  - “.gitignore”
- Local vs Remote
Creating Git Repository

- Initialize a new repository
  
  `git init`

- Clone an existing repository
  
  `git clone pathToRepo`

Specify Remotes

  `git remote add origin pathToRepo`
Steps to Setup Tracking

● Locate the folder/files

● Initialize an empty repository
  
  git init

● Declare “.gitignore”

● Add files to index
  
  git add folder1 file1 file3
  git .

● Commit with descriptive message
  
  git commit -m "Initial configuration"
Backup / Restore Configurations

- **Backup configurations**
  
  ```
  git clone --bare . /share/configs.git
  git remote add origin /share/configs.git
  ```

- **Restore to a new location – by cloning**
  
  ```
  cd /new-location-path
  git clone /share/configs.git
  git checkout specify-commit
  ```
Auto Propagating/Updating

- Propagated configurations
  - `git pull`
- Selective Updates
  - When / What
- Auto-propagate
  - `cron job`
Handling Large Repositories

- Single Repository (Not Recommended)
- Main Repository with Multiple Submodules
  - Clear dependency
    - Embedding Software Libraries
    - Submodules/Frameworks (e.g. Android SDK)
- Many Repositories
  - Loose Dependency
  - Need to use a wrapper script
    (e.g. Google “repo” script)
Usage Samples

- “/etc” folder
- Apache web server configurations
- Jenkins configurations
- Tool configurations
- Subversion configuration files
- Compilers – different versions tracked in 1 repository
- Eclipse configurations
- Android SDK
Backup Options

- Keep Local
  - Less critical
  - No need to propagate
- Push to share drive
  - Need to share with other servers, etc.
- Push to Git server
  - Configurations critical for compliance, etc.
Demo
Tools

- Git Command Line
- Gitk
- SmartGit (All Platforms)
- SourceTree (Mac)
- Meld (Compare Tool)
Conclusions

- Git to track configurations
- Git allows backup/rollback of configurations
- Git allows propagation of configurations
- Track embedded configurations
- Track modifications by junior administrators!
Contact Information

Anas Mughal
anas@devsenesce.com
@anasmughal

http://www.linkedin.com/in/anasmughal
http://anas-mughal.com