4.1 Protokoly

Git může k přenosu dat používat jeden ze čtyř hlavních síťových protokolů: Local, Secure Shell (SSH), Git nebo HTTP. V této části se podíváme na to, co jsou jednotlivé protokoly zač a za jakých okolností je (ne)vhodné je použít.

Neměli bychom zamlčet ani to, že s výjimkou protokolu HTTP všechny vyžadují, aby byl na serveru nainstalován a spuštěn systém Git.

4.1.1 Protokol Local

Nejzákladnější variantou je *protokol Local*, v němž je vzdálený repozitář uložen v jiném adresáři na disku. Často se využívá v případech, kdy mají všichni z vašeho týmu přístup k vašim sdíleným souborům, např. přes připojení systému NFS, nebo – v méně pravděpodobném případě – se všichni přihlašují na jednom počítači. Tato druhá varianta není právě ideální, protože všechny instance repozitáře s kódem jsou v takovém případě umístěny v jednom počítači, čímž se zvyšuje riziko nevratné ztráty dat.

Máte-li připojený sdílený systém souborů, můžete klonovat, odesílat a stahovat z lokálního souborového repozitáře (local file-based repository). Chcete-li takový repozitář naklonovat nebo přidat jako vzdálený repozitář do existujícího projektu, použijte jako URL cestu k repozitáři. K naklonování lokálního repozitáře můžete použít příkaz například v tomto tvaru:

```
$ git clone /opt/git/project.git
```

Nebo můžete provést následující:

```
$ git clone file:///opt/git/project.git
```

Pokud na začátek URL explicitně zadáte výraz `file://`, pracuje Git trochu jinak. Pokud pouze zadáte cestu, Git se pokusí použít hardlinky nebo rovnou zkopírovat soubory, které potřebuje. Pokud zadáte výraz `file://`, Git spustí procesy, jež běžně používá k přenosu dat prostřednictvím sítě. Síť je většinou výrazně méně výkonnou metodou přenosu dat. Hlavním důvodem, proč zadat předponu `file://` je to, že tak získáte čistou kopii repozitáře bez nepotřebných referencí a objektů, např. po importu z jiného verzovacího systému a podobně (úkony správy jsou popsány v kapitole 9). My budeme používat normální cestu, neboť tato metoda je téměř vždy rychlejší.

Kap.

K přidání lokálního repozitáře do existujícího projektu Git můžete použít příkaz například v tomto tvaru:

```
$ git remote add local_proj /opt/git/project.git
```

Poté můžete odesílat data a stahovat je z tohoto vzdáleného serveru, jako byste tak činili prostřednictvím sítě.

Výhody

Výhoda souborových repozitářů spočívá v tom, že jsou jednoduché a používají existující oprávnění k souborům a síťový přístup. Pokud už máte sdílený systém souborů, k němuž má přístup celý váš tým, je nastavení repozitáře velice jednoduché. Kopii holého repozitáře umístíte někam, kam mají všichni sdílený přístup, a nastavíte oprávnění ke čtení/zápisu stejně jako u jakéhokoli jiného sdíleného adresáře. O exportu kopie holého repozitáře pro tento účel se více dočtete v následující části „Jak umístit Git na server“.

Jedná se také o výbornou možnost, jak rychle získat práci z pracovního repozitáře někoho jiného. Pokud vy a váš kolega pracujete na společném projektu a vy potřebujete provést checkout kolegových dat, bývá například příkaz `git pull /home/john/project` jednodušší než odesílat data na vzdálený server a odsud je opět stahovat.

Nevýhody

Nevýhodou této metody je, že nastavit a získat sdílený přístup z více umístění je většinou těžší než obyčejný síťový přístup. Budete-li chtít pracovat doma a odeslat data z notebooku, budete muset připojit vzdálený disk, což může být ve srovnání s přístupem prostřednictvím sítě složité a pomalé.

Zapomenout bychom neměli ani na to, že používáte-li sdílené připojení určitého druhu, nemusí být tato možnost vždy nutně nejrychlejší. Lokální repozitář je rychlý pouze v případě, že máte rychlý přístup k datům. Repozitář na NFS je často pomalejší než repozitář nad SSH na tomtéž serveru, který ve všech systémech umožňuje spustit Git z lokálních disků.

4.1.2 Protokol SSH

Patrně nejčastějším přenosovým protokolem pro systém Git je SSH. Je to z toho důvodu, že SSH přístup k serverům je na většině míst už nastaven, a pokud ne, není ho těžké nastavit. SSH je navíc jediným síťovým protokolem, z nějž lze snadno číst a do nějž lze snadno zapisovat. Oba zbývající síťové protokoly (HTTP i Git) jsou většinou určeny pouze ke čtení, a proto i když je máte k dispozici, budete potřebovat SSH protokol pro příkazy k zápisu. SSH je také síťovým protokolem s ověřováním, a protože je hojně rozšířen, je jeho nastavení a používání většinou snadné.

Chcete-li naklonovat repozitář Git pomocí protokolu SSH, zadejte „ssh:// URL“, například:

```
$ git clone ssh://user@server:project.git
```

Protokol ostatně ani nemusíte zadávat – pokud žádný výslovně neurčíte, Git použije SSH jako výchozí možnost:

```
$ git clone user@server:project.git
```

Stejně tak nemusíte zadávat ani uživatele, Git automaticky použije uživatele, jehož účtem jste právě přihlášeni.

Výhody

Používání protokolu SSH přináší mnoho výhod. Především byste ho měli používat vždy, když chcete v síti ověřovat oprávnění k zápisu do repozitáře. Zadruhé: protokol SSH má snadné nastavení – SSH démoni jsou zcela běžní, správci sítě si s nimi většinou vědí rady a mnoho distribucí OS je má ve výchozí instalaci nebo má nástroje, aby s nimi mohly pracovat. Z dalších výhod bychom měli zmínit také to, že přístup přes protokol SSH je bezpečný, veškerý přenos dat je šifrovaný a ověřený. A stejně jako protokoly Git a Local je i protokol SSH výkonný. Data jsou před přenosem upravena do co nejkompaktnější podoby.

Nevýhody

Nevýhodou protokolu SSH je, že neumožňuje anonymní přístup do repozitáře. Chce-li někdo získat přístup do vašeho repozitáře, byť třeba jen ke čtení, musí mít přístup k vašemu počítači přes SSH. Proto se protokol SSH nehodí pro projekty s otevřeným zdrojovým kódem. Pokud repozitář používáte jen v rámci firemní sítě, bude pro vás protokol SSH zřejmě naprosto ideální. Pokud chcete povolit anonymní přístup pro čtení k vašim projektům, budete muset nastavit protokol SSH k odesílání svých dat, ale přidat jiný protokol, pomocí nějž budou ostatní tato data stahovat.

4.1.3 Protokol Git

Dalším protokolem v pořadí je protokol Git. Je to speciální démon, který je distribuován spolu se systémem Git. Naslouchá na vyhrazeném portu (9418) a poskytuje podobnou službu jako protokol SSH, avšak bez jakéhokoli ověřování. Chcete-li, aby byl repozitář obsluhován protokolem Git, musíte vytvořit soubor `git-export-daemon-ok` – démon nebude repozitář obsluhovat, dokud v něm tento soubor nebude. Žádné jiné zabezpečení k dispozici není. Repozitář Git je buď dostupný pro všechny a všichni z něj mohou klonovat, nebo dostupný není. To znamená, že se přes tento protokol nedají odesílat žádné revize. Možnost odesílání lze aktivovat, ale vzhledem k tomu, že protokol neumožňuje ověřování, aktivované odesílání znamená, že kdokoli na internetu, kdo najde URL vašeho projektu, do něj bude moci odesílat data. Tato možnost se však téměř nepoužívá.

Výhody

Protokol Git je ze všech dostupných protokolů nejrychlejší. Potřebujete-li, aby protokol obsluhoval frekventovaný provoz u veřejného projektu nebo velmi velký projekt, u nějž není třeba ověřování identity uživatele ohledně oprávnění pro čtení, bude k obsluze nejvhodnější pravděpodobně právě démon Git.

Používá stejný mechanismus přenosu dat jako protokol SSH, na rozdíl od něj ale není zpomalován šifrováním a ověřováním.

Nevýhody

Nevýhodou protokolu Git je, že neprovádí ověřování. Většinou není žádoucí, aby protokol Git tvořil jediný přístup k vašemu projektu. Protokol Git většinou využijete v kombinaci s přístupem přes SSH. Protokol SSH bude nastaven pro několik málo vývojářů s oprávněním k zápisu (odesílání dat) a všichni ostatní budou používat `git://` pro přístup pouze ke čtení. Pravděpodobně se také jedná o protokol s nejobtížnějším nastavením. Vyžaduje spuštění vlastního démona – na jeho nastavení se podíváme v části „Gitosis“ této kapitoly – a dále konfiguraci `xinetd` nebo podobnou, která také není právě jednoduchá. Vyžaduje rovněž povolení přístupu k portu 9418 skrz firewall. Tento port nepatří mezi standardní porty, které by firemní firewally vždy povolovaly. Velkými podnikovými firewally je tento málo rozšířený port většinou blokován.

4.1.4 Protokol HTTP/S

Na konec jsme si nechali protokol HTTP. Co je na protokolu HTTP nebo HTTPS sympatické, je jejich jednoduché nastavení. Jediné, co většinou stačí udělat, je umístit holý repozitář Git do kořenového adresáře HTTP a nastavit příslušný zásuvný modul `post-update` (zásuvné moduly Git viz kapitola 7). Tím je nastavení hotové. V tuto chvíli může každý, kdo má přístup na webový server, kam jste repozitář uložili, tento repozitář naklonovat. Chcete-li u svého repozitáře nastavit oprávnění pro čtení pomocí protokolu HTTP, proveďte následující:

Kap.

```
$ cd /var/www/htdocs/
$ git clone --bare /path/to/git_project gitproject.git
$ cd gitproject.git
$ mv hooks/post-update.sample hooks/post-update
$ chmod a+x hooks/post-update
```

A to je vše. Zásuvný modul `post-update`, který je standardně součástí systému Git, spustí příkaz `git update-server-info`, který zajistí správné vyzvedávání a klonování dat přes protokol HTTP. Tento příkaz se spustí, když do tohoto repozitáře odesíláte data přes protokol SSH. Ostatní mohou klonovat třeba takto:

```
$ git clone http://example.com/gitproject.git
```