Comunicado Técnico 122
ISSN 1677-8464
Junho, 2016
Campinas, SP

Implementando um servidor Git para acesso através de HTTPS

Fonte: https://git-scm.com/

Jorge Luiz Corrêa[1]

O Git

O Git é um sistema de controle de versões distribuído. Ele foi inicialmente proposto por Linus Torvalds em 2005 para utilização no desenvolvimento do Kernel Linux. O que o diferencia de sistemas como o Subversion é justamente sua característica de distribuição. Não existe local central, tomado como matriz, usado pelos clientes para sincronizar o conteúdo. Um diretório versionado com Git representa o histórico completo dos arquivos, em cada local, ou seja, se existirem 2 clientes e um servidor, cada um deles terá o histórico completo dos dados, não apenas o servidor. Desta forma, evita-se ter um ponto central (servidor) onde os arquivos são tomados como base para comparações, controle e distribuição entre os adquirentes. O servidor atua no papel de um repositório de dados, um nó a mais para distribuição dos arquivos.

Em sua operação normal o Git trabalha em um diretório do sistema de arquivos, de modo que o usuário indica os arquivos que serão versionados, podendo incluí-los para o versionamento e assim comparando-o com outras versões em repositórios remotos.

Neste documento o foco é na implementação do acesso a um repositório (servidor) Git através do protocolo HTTPS, ou seja, via web com criptografia. Esta é uma alternativa bastante comum visto que este protocolo dificilmente é filtrado em ambientes corporativos, permitindo então que o servidor seja acessado sem grandes problemas com a filtragem de pacotes.

WebDAV

O Web Distributed Authoring and Versioning (WebDAV) é uma extensão do protocolo HTTP que inclui novas possibilidades (diferentes das utilizadas no HTTP como GET, POST e afins), do usuário alterar um conteúdo via web. O WebDAV está definido no (DUSSEAULT, 2007).

As primeiras implementações de um servidor Git para uso via HTTP(S) eram realizadas pela utilização do WebDAV. Desta forma, um servidor web, como o Apache, era configurado com um módulo WebDAV que permitia a manipulação de áreas do sistema de arquivos do servidor. Os repositórios Git eram criados em sua forma normal, no sistema de arquivos, e o WebDAV funcionava como uma interface entre o acesso web via HTTP(S) e os arquivos que deveriam ser manipulados.

[1] Cientista da Computação, mestre em Ciência da Computação, analista da Embrapa Informática Agropecuária, Campinas, SP.

Embrapa

Em versões mais recentes do Git, a equipe de desenvolvimento deixou de indicar a implementação por meio do WebDAV e disponibilizou um *script* CGI como nova recomendação para implementações de servidores para Git. Este *script* é o git-http-backend.

O git-http-backend

O git-http-backend é uma implementação do Git para o lado servidor, pelo protocolo HTTP(S). Conforme a própria descrição do *script*, o git-http-backend' é um programa CGI para servir conteúdo de repositórios Git, para clientes Git, acessando os repositórios pelos protocolos HTTP e HTTPS' (GIT MAN…, 2015).

O git-http-backend depende de configurações do servidor web para que opere satisfatoriamente. O restante deste documento explana sobre os detalhes técnicos de uma instalação baseada no sistema operacional Ubuntu Server 14.04 LTS. No entanto, as configurações do Apache são praticamente as mesmas em outros sistemas *NIX. As maiores mudanças são na forma como cada sistema organiza seus arquivos de configuração e suas localizações no sistema de arquivos.

Instalando o Git

A instalação do Git pode ser realizada pelo sistema APT conforme comando a seguir. Todos os comandos deste documento devem ser executados como root.

```
apt-get install git
```

Este comando instalará os binários, manuais e dependências para utilização do Git.

Instalando o Apache

O Apache deve ser instalado com o módulo de CGI-BIN.

```
apt-get install apache2 libapache2-mod-fcgid
```

Após a instalação, o módulo do CGI deve ser habilitado. Habilitar também o módulo 'alias' que será utilizado na configuração.

```
a2enmod fcgid alias
service apache2 restart
```

Criando o diretório de repositórios

Os repositórios residirão no sistema de arquivos no caminho /var/git/. Cada diretório dentro deste caminho representará um repositório Git. Alterar as permissões desse diretório para o usuário que executa o Apache, pois o servidor web deverá escrever neste diretório.

```
mkdir -p /var/git
chown -R www-data.www-data /var/git
```

Configurando o Apache

O servidor sendo utilizado será dedicado totalmente ao serviço Git. Esta escolha é opcional e depende de como a infraestrutura é organizada. Utilizar serviços em servidores dedicados facilita a manutenção tanto do serviço quanto do servidor. Assim sendo, o Apache responderá apenas pelo site de repositórios. Existirão, portanto, dois arquivos de configuração de sites sendo um para o acesso em HTTP e outro para o acesso em HTTPS, de forma que o HTTP é um redirecionamento para o HTTPS, forçando que todos os acessos sejam criptografados. Observação: caso a opção seja por um servidor compartilhado, será necessário criar um *virtualhost* para o sistema, de forma semelhante aos que já existam.

Outra característica importante do servidor em implementação é a utilização de usuários de uma base LDAP. Todos os acessos a algum dos repositórios apenas serão permitidos para usuários autenticados nesta base. Além disso, a autorização ocorrerá por repositório e um grupo LDAP. Recomenda-se que, para cada repositório criado, exista um grupo no LDAP cujos membros são autorizados. Este grupo será declarado na configuração do Apache. A instalação e a configuração de um servidor de diretórios LDAP está fora do escopo deste texto.

Arquivo de configuração do site *default* do Apache

O arquivo /etc/apache2/sites-available/000-default.conf deve conter:

```
<VirtualHost *:80>
 Redirect permanent / https://..FQDN..
</VirtualHost>
```

A diretiva Redirect força que o navegador do cliente faça uma nova solicitação web, porém, no site com SSL habilitado (HTTPS). A URL https://..FQDN.. é o nome completo do servidor sendo implementado (seja o nome do *host* ou um ALIAS no DNS), porém, utilizando o protocolo HTTPS (FQDN = *Fully Qualified Domain Name*).

Arquivo de configuração para o *virtualhost* HTTPS

Para utilizar algum *virtualhost* com SSL deve-se primeiramente habilitar o módulo correspondente.

```
a2enmod ssl
service apache2 restart
```

Uma vez habilitado, cria-se o arquivo de configuração /etc/apache2/sites-available/git-ssl-http-backend.conf com o seguinte conteúdo:

```
<IfModule mod_ssl.c>
<VirtualHost _default_:443>
 ServerAdmin email@dominio
 ServerName hostname.dominio

 # DocumentRoot e' necessario no virtualhost pois o git-http-backend concatena este valor
 # para determinar onde o repositorio esta no disco. Caso se utilize um local diferente
 # do /var/www, deve-se informar, caso contrario apache utilizara as configuracoes default
 # e considerara que os repositorios estao em /var/www.
 DocumentRoot /var/git

 # Variaves necessarias para o git-http-backend
 #
 # GIT_HTTP_EXPORT_ALL indica que todos os diretorios do GIT_PROJECT_ROOT podem ser
 #   exportados. Caso nao seja configurada, cada repositorio devera ter o arquivo
 #   git-daemon-export-ok.
 # GIT_PROJECT_ROOT indica a raiz onde estao os repositorios.
 # Scriptalias e' uma diretiva do Apache que indica que o primeiro caminho deve ser mapeado
 # para o script CGI do segundo caminho. Dessa forma, qualquer acesso (/) sera atendido pelo
 # script /usr/lib/git-core/git-http-backend/. E' importante notar a / no final pois o que
 # vier na URL (por exemplo, /repositorio.git) sera concatenado na URL para o CGI, na
 # forma /usr/lib/git-core/git-http-backend/repositorio.git. Se o / nao for informado o
 # Apache recebera um acesso que sera mapeado para /usr/lib/git-core/git-http-
 #   backendrepositorio.git.
 SetEnv GIT_HTTP_EXPORT_ALL
 SetEnv GIT_PROJECT_ROOT /var/git
 ScriptAlias / /usr/lib/git-core/git-http-backend/

 # Permitir ao Apache executar scripts CGI no diretorio /usr/lib/git-core/, onde se encontra
 #   o git-http-backend.
 <Directory "/usr/lib/git-core/">
      Options +ExecCGI
 </Directory>

 # Para o DocumentRoot, negar todo tipo de acesso.
 <Directory />
      SSLRequireSSL
      AllowOverride All
      Require all denied
 </Directory>

 # Para controlar o acesso aos repositorios, eles devem ser declarados como um Location,
 # pois se trata de um acesso em https://dominio/nomerepositorio.git e, embora exista o
 # diretorio nomerepositorio.git no GIT_PROJECT_ROOT, esta solicitacao e' atendida pelo
 # script git-http-backend. Caso se utilize a diretiva Directory, não é possivel declarar as
 # autenticacoes dos usuarios, pois na verdade o diretorio nunca e' acessado.
 #
 # Na configuracao de uma Location conforme abaixo, e' requerida autenticacao tanto para
 # leitura quanto escrita.
 #
      # No caso de uma leitura o git-http-backend tem o comportamento padrao de aceitar sem
 # autenticacao.
 #
 # No caso de uma escrita o git-http-backend precisa que variaveis de autenticacao sejam
 # setadas durante o acesso. Desta forma, deve-se utilizar o AuthType para que o apache
 # autentique o usuario de alguma forma (neste caso e' via LDAP), exporte essas informacoes
 # e o script prossiga normalmente para escrita. Sem essa autenticacao e' necessario alterar
 # a configuracao http.receivepack de cada repositorio, de forma que ele aceite escrita sem
 # autenticacao.
 <Location /teste.git>
```

```
       AuthType Basic
       AuthName "Repositorio Git do dominio"
       AuthBasicProvider ldap
       AuthLDAPURL "ldaps://SERVIDOR_LDAP:636/DC=d
ominio,DC=br?uid?sub?(objectClass=*)"
       # AuthLDAPGroupAttributeIsDN off indica que
vai procurar apenas o id do usuario
       #   como membro do grupo, nao o dn inteiro.
       AuthLDAPGroupAttributeIsDN off
       # AuthLDAPGroupAttribute memberUid indica
que este campo sera usado para procurar
       #   um membro de grupo (depende de como o
LDAP e' configurado).
       AuthLDAPGroupAttribute memberUid
       Options ExecCGI FollowSymLinks Indexes
       <RequireAny>
          Require ldap-group cn=gi_teste,ou=Group
,dc=cnptia,dc=embrapa,dc=br
       </RequireAny>
       SSLRequireSSL
 </Location>

 ErrorLog ${APACHE_LOG_DIR}/git_error.log

 LogLevel warn

 CustomLog ${APACHE_LOG_DIR}/git_ssl_access.log
combined
 SSLEngine on
 SSLCertificateFile    /etc/ssl/certs/ssl-cert-
-snakeoil.pem
       SSLCertificateKeyFile /etc/ssl/private/ssl-
-cert-snakeoil.key

 <FilesMatch "\.(cgi|shtml|phtml|php)$">
       SSLOptions +StdEnvVars
 </FilesMatch>
 <Directory /usr/lib/cgi-bin>
       SSLOptions +StdEnvVars
 </Directory>
 BrowserMatch "MSIE [2-6]" \
       nokeepalive ssl-unclean-shutdown \
       downgrade-1.0 force-response-1.0
 # MSIE 7 and newer should be able to use keepa-
live
 BrowserMatch "MSIE [17-9]" ssl-unclean-shutdown

</VirtualHost>
</IfModule>
```

Embora o arquivo esteja comentado, algumas considerações são importantes em face das diretivas utilizadas.

O *script* CGI, sendo utilizado, é instalado no caminho /usr/lib/git-core/git-http-backend. Este *script* recebe parâmetros para executar suas ações. Um dos parâmetros importantes nesta configuração é o DocumentRoot do *virtualhost*, configurado neste caso para /var/git. Este parâmetro é utilizado pelo *script* para determinar onde os arquivos estão no sistema de arquivos. Ele é concatenado com valores tais como o nome do repositório, obtido da URL de acesso, para localizar os arquivos do repositório.

A variável GIT_PROJECT_ROOT indica o diretório raiz onde estão os repositórios. A variável GIT_HTTP_EXPORT_ALL indica ao *script* que todos os diretórios dentro de GIT_PROJECT_ROOT podem ser exportados através deste *backend*. Conforme mencionado, o Git trabalha diretamente com o sistema de arquivos. Para que exista um controle sobre quais repositórios no sistema de arquivos podem ser acessados através do http-backend, o *script* foi desenvolvido de modo a verificar a existência de um arquivo chamado 'git-daemon-export-ok' dentro do repositório. Caso ele exista significa que o *script* CGI pode acessar este repositório. Caso não, trata-se de um repositório que só pode ser acessado diretamente pelo sistema de arquivos, não podendo ser usado via HTTP(S). Uma vez que este servidor é dedicado aos repositórios Git, a variável GIT_HTTP_EXPORT_ALL é uma maneira de liberar o acesso aos repositórios sem que seja necessário criar este arquivo dentro de cada um deles.

A diretiva ScriptAlias do Apache é utilizada quando se deseja mapear o acesso a determinada parte do site para um *script* CGI. No caso da configuração em questão, qualquer acesso (representado pelo /) é mapeado para o *script* /usr/lib/git-core/git-http-backend/. É possível observar que existe uma barra (/) ao final do caminho, embora git-http-backend seja um arquivo. Esta é a maneira utilizada para passar parâmetros ao *script*. Desta forma, se o acesso vier para https://..FQDN../nome_repositorio, será passado ao *script* da forma /usr/lib/git-core/git-http-backend/nome_repositorio. O *script* fará então as concatenações com o DocumentRoot e localizará os arquivos deste repositório, no sistema de arquivos.

A diretiva <Directory "/usr/lib/git-core/"> do arquivo de configuração anterior é utilizada para instruir o Apache a executar *scripts* CGI contidos em uma localização do sistema de arquivos. O Directory é um meio de permitir e configurar acessos do Apache a partes do sistema de arquivos que estão fora do DocumentRoot do *virtualhost* em questão.

De forma semelhante, a configuração <Directory /> mostrada é utilizada para negar qualquer tipo de

acesso. Esta é a raiz do *virtualhost*, ou seja, o /var/git. Uma vez que não é permitida a listagem de conteúdo desta localização, ela deve ser explicitamente negada. Os acessos apenas serão permitidos para URLs que contenham o nome de algum repositório.

Cada repositório deverá ser declarado no arquivo de configuração com uma diretiva Location. Com esta diretiva é possível aplicar configurações para a localização em questão, sendo que esta localização refere-se a uma parte da URL acessada pelo usuário. Por exemplo, as configurações declaradas em <Location /teste.git> serão utilizadas quando um cliente acessar https://..FQDN../teste.git.

O *script* git-http-backend possui alguns comportamentos padrões quanto às autorizações de acesso. Quando habilitado ele permite que um repositório (Location) seja acessado para leitura sem a necessidade de autenticação. Isso permitirá, por exemplo, as ações *git fetch*, *git pull*, e *git clone*. No entanto, para ações de escritas, é necessário que variáveis de ambiente estejam habilitadas durante o acesso. Algumas dessas variáveis são configuradas automaticamente por módulos do Apache. Assim, para permitir que os repositórios aceitem escritas, como a ação *git push*, se faz necessário a configuração de algum método de autenticação. No caso deste tutorial, o arquivo prepara uma autenticação utilizando um servidor LDAP.

Para habilitar os módulos de autenticação do Apache deve-se utilizar:

```
a2enmod authn_core authz_core ldap
service apache2 restart
```

O exemplo a seguir mostra uma configuração simples para o caso de usuários mantidos em um arquivo. As diretivas “Auth” criam as variáveis necessárias para que o *script* CGI permita o acesso aos repositórios quando o processo de autenticação finalizar com sucesso (o *script* CGI necessita que as variáveis existam para iniciar suas ações, pois sem elas, ele não executa; além da existência de algum tipo de autenticação, deve ficar óbvio que o processo de autenticação em si deve ocorrer normalmente e apenas os usuários autenticados poderão acessar os dados).

```
<Location />
 AuthType Digest
 AuthName "Shared Git Repo"
 AuthUserFile /var/git/.htpasswd
 Require valid-user
</Location>
```

Voltando à configuração que utiliza (ZEILENGA, 2006) a diretiva AuthBasicProvider habilita o módulo 'ldap' que fará a autenticação e AuthLDAPURL informa o servidor LDAP a ser utilizado, bem como o caminho dentro do diretório LDAP (Base DN) onde os objetos de usuário estão armazenados.

A diretiva 'AuthLDAPGroupAttributeIsDN off' indica que o Apache deve considerar como membro de um grupo apenas o id do usuário, não seu DN inteiro. Esta configuração é compatível com um diretório LDAP organizado da forma que grupos contenham usuários e esta informação seja baseada apenas no id dos usuários. Já a diretiva 'AuthLDAPGroupAttribute memberUid' indica que o campo do objeto grupo no diretório LDAP, a ser utilizado para definir que um usuário é membro deste grupo, é o memberUid. Ou seja, o diretório LDAP deve conter grupos em que existam uma ou mais entradas MemberUid indicando quais usuários são integrantes dele.

O bloco <RequireAny> informa quais informações são necessárias para que o processo de autenticação seja considerado satisfatório. Será necessário existir um usuário (Require valid-user) e este pertencer ao grupo LDAP informado (Require ldap-group). Somente assim o acesso ao repositório será liberado.

Após a criação do arquivo de configuração, o site deverá ser habilitado no Apache:

```
a2ensite git-ssl-http-backend
service apache2 restart
```

Criando um repositório

Para a criação de um novo repositório, deve-se utilizar os seguintes comandos:

```
cd /var/git
mkdir repositorio.git
cd repositorio.git
git init --bare
git --bare update-server-info
chown -R www-data. /var/git/repositorio.git
```

A linha 'git init --bare' é a que, de fato, cria o repositório. O parâmetro --bare faz com que os arquivos necessários para o controle e versionamento sejam criados dentro do próprio diretório corrente. O padrão do Git, ao inicializar um repositório, é criar um diretório oculto chamado .git. A linha '--bare update-server-info' cria e/

ou atualiza determinados arquivos, como: objects/info/packs e info/refs, sendo necessária quando o repositório será acessado via HTTP(S).

Após a criação do repositório no sistema de arquivos, deve-se criar uma entrada de configuração no arquivo /etc/apache2/sites-available/git-ssl-http-backend.conf conforme a seguir:

```
<Location /repositorio.git>
 AuthType Basic
 AuthName "Repositorio Git do Dominio"
 AuthBasicProvider ldap
 AuthLDAPURL "ldaps://ldapserver.dominio:636/DC=dominio,DC=br?uid?sub?(objectClass=*)"
 AuthLDAPGroupAttributeIsDN off
 AuthLDAPGroupAttribute memberUid
 Options ExecCGI FollowSymLinks Indexes
 <RequireAny>
   Require ldap-group cn=gi_repositoriogit,ou=Group,dc=dominio,dc=br
 </RequireAny>
 SSLRequireSSL
<Location>
```

Deve-se reiniciar o Apache para que as modificações tenham efeito:

```
service apache2 restart
```

Testando um repositório

Para teste do servidor, supondo que o repositório repositorio.git foi criado, pode-se utilizar os seguintes comandos, de preferência em um *host* diferente do servidor para que o acesso via rede seja testado.

```
git clone https://endereco.servidor.dominio/repositorio.git repositorio.git
cd repositorio.git/
echo "conteudo." > arquivo1.txt
git add arquivo1.txt
git commit -m "Commit arquivo1.txt"
git push origin master
```

Referências

DUSSEAULT, L.(Ed.). **RFC 4918**: Web distributed authoring and versioning (WebDAV) . 2015. Disponível em: <https://tools.ietf.org/html/rfc2518>. Acesso em: 11 dez. 2015.

GIT MAN git-http-backend manual page. 2015. Disponível em: <https://www.kernel.org/pub/software/scm/git/docs/git-http-backend.html>. Acesso em: 11 dez. 2015.

ZEILENGA, K. (Ed.). **Lightweight Directory Access Protocol (LDAP)**: technical specification road map. 2006. Disponível em: <https://tools.ietf.org/html/rfc4510>. Acesso em: 7 mar. 2016.

Comunicado Técnico, 122

Embrapa Informática Agropecuária
Endereço: Caixa Postal 6041 - Barão Geraldo
13083-886 - Campinas, SP
Fone: (19) 3211-5700
www.embrapa.br/informatica-agropecuaria
sac: www.embrapa.br/fale-conosco/sac/

MINISTÉRIO DA AGRICULTURA, PECUÁRIA E ABASTECIMENTO

1ª edição publicação digital - 2016

Expediente

Supervisão editorial: *Stanley Robson de Medeiros Oliveira, Neide Makiko Furukawa*

Normalização bibliográfica: *Maria Goretti Gurgel Praxedes*

Revisão de texto: *Adriana Farah Gonzalez*

Editoração eletrônica: *Neide Makiko Furukawa*

CGPE 12903