NAME

git-daemon - A really simple server for Git repositories

SYNOPSIS

git daemon [--verbose] [--syslog] [--export-all]
[--timeout=<n>] [--init-timeout=<n>] [--max-connections=<n>]
[--strict-paths] [--base-path=<path>] [--base-path-relaxed]
[--user-path | --user-path=<path>]
[--interpolated-path=<pathtemplate>]
[--reuseaddr] [--detach] [--pid-file=<file>]
[--enable=<service>] [--disable=<service>]
[--allow-override=<service>] [--forbid-override=<service>]
[--access-hook=<path>] [--[no-]informative-errors]
[--inetd |
[--listen=<host_or_ipaddr>] [--port=<n>]
[--user=<user> [--group=<group>]]]
[--log-destination=(stderr|syslog|none)]
[<directory>...]

DESCRIPTION

A really simple TCP Git daemon that normally listens on port "DEFAULT_GIT_PORT" aka 9418. It waits for a connection asking for a service, and will serve that service if it is enabled.

It verifies that the directory has the magic file "git-daemon-export-ok", and it will refuse to export any Git directory that hasn't explicitly been marked for export this way (unless the **--export-all** parameter is specified). If you pass some directory paths as *git daemon* arguments, the offers are limited to repositories within those directories.

By default, only **upload-pack** service is enabled, which serves *git fetch-pack* and *git ls-remote* clients, which are invoked from *git fetch*, *git pull*, and *git clone*.

This is ideally suited for read-only updates, i.e., pulling from Git repositories.

An **upload-archive** also exists to serve *git archive*.

OPTIONS

--strict-paths

Match paths exactly (i.e. don't allow "/foo/repo" when the real path is "/foo/repo.git" or "/foo/repo/.git") and don't do user-relative paths. *git daemon* will refuse to start when this option

is enabled and no directory arguments are provided.

--base-path=<path>
Remap all the path requests as relative to the given path. This is sort of "Git root" - if you run *git daemon* with *--base-path=/srv/git* on example.com, then if you later try to pull *git://example.com/hello.git, git daemon* will interpret the path as **/srv/git/hello.git**.

--base-path-relaxed
If --base-path is enabled and repo lookup fails, with this option *git daemon* will attempt to lookup without prefixing the base path. This is useful for switching to --base-path usage, while still allowing the old paths.

--interpolated-path=<pathtemplate>
To support virtual hosting, an interpolated path template can be used to dynamically construct alternate paths. The template supports %H for the target hostname as supplied by the client but converted to all lowercase, %CH for the canonical hostname, %IP for the server's IP address, %P for the port number, and %D for the absolute path of the named repository. After interpolation, the path is validated against the directory list.

--export-all
Allow pulling from all directories that look like Git repositories (have the *objects* and *refs* subdirectories), even if they do not have the *git-daemon-export-ok* file.

--inetd
Have the server run as an inetd service. Implies --syslog (may be overridden with **--log-destination=**). Incompatible with --detach, --port, --listen, --user and --group options.

--listen=<host_or_ipaddr>
Listen on a specific IP address or hostname. IP addresses can be either an IPv4 address or an IPv6 address if supported. If IPv6 is not supported, then --listen=hostname is also not supported and --listen must be given an IPv4 address. Can be given more than once. Incompatible with **--inetd** option.

--port=<n>
Listen on an alternative port. Incompatible with **--inetd** option.

--init-timeout=<n>
Timeout (in seconds) between the moment the connection is established and the client request is received (typically a rather low value, since that should be basically immediate).

--timeout=<n>
: Timeout (in seconds) for specific client sub-requests. This includes the time it takes for the server to process the sub-request and the time spent waiting for the next client's request.

--max-connections=<n>
: Maximum number of concurrent clients, defaults to 32. Set it to zero for no limit.

--syslog
: Short for **--log-destination=syslog**.

--log-destination=<destination>
: Send log messages to the specified destination. Note that this option does not imply --verbose, thus by default only error conditions will be logged. The <destination> must be one of:

  stderr
  : Write to standard error. Note that if **--detach** is specified, the process disconnects from the real standard error, making this destination effectively equivalent to **none**.

  syslog
  : Write to syslog, using the **git-daemon** identifier.

  none
  : Disable all logging.

  The default destination is **syslog** if **--inetd** or **--detach** is specified, otherwise **stderr**.

--user-path, --user-path=<path>
: Allow ~user notation to be used in requests. When specified with no parameter, requests to git://host/~alice/foo is taken as a request to access *foo* repository in the home directory of user **alice**. If **--user-path=path** is specified, the same request is taken as a request to access **path/foo** repository in the home directory of user **alice**.

--verbose
: Log details about the incoming connections and requested files.

--reuseaddr
: Use SO_REUSEADDR when binding the listening socket. This allows the server to restart without waiting for old connections to time out.

--detach

Detach from the shell. Implies --syslog.

--pid-file=<file>
Save the process id in *file*. Ignored when the daemon is run under **--inetd**.

--user=<user>, --group=<group>
Change daemon's uid and gid before entering the service loop. When only **--user** is given without **--group**, the primary group ID for the user is used. The values of the option are given to **getpwnam(3)** and **getgrnam(3)** and numeric IDs are not supported.

Giving these options is an error when used with **--inetd**; use the facility of inet daemon to achieve the same before spawning *git daemon* if needed.

Like many programs that switch user id, the daemon does not reset environment variables such as **$HOME** when it runs git programs, e.g. **upload-pack** and **receive-pack**. When using this option, you may also want to set and export **HOME** to point at the home directory of **<user>** before starting the daemon, and make sure any Git configuration files in that directory are readable by **<user>**.

--enable=<service>, --disable=<service>
Enable/disable the service site-wide per default. Note that a service disabled site-wide can still be enabled per repository if it is marked overridable and the repository enables the service with a configuration item.

--allow-override=<service>, --forbid-override=<service>
Allow/forbid overriding the site-wide default with per repository configuration. By default, all the services may be overridden.

--[no-]informative-errors
When informative errors are turned on, git-daemon will report more verbose errors to the client, differentiating conditions like "no such repository" from "repository not exported". This is more convenient for clients, but may leak information about the existence of unexported repositories. When informative errors are not enabled, all errors report "access denied" to the client. The default is --no-informative-errors.

--access-hook=<path>
Every time a client connects, first run an external command specified by the <path> with service name (e.g. "upload-pack"), path to the repository, hostname (%H), canonical hostname (%CH), IP address (%IP), and TCP port (%P) as its command-line arguments. The external command can decide to decline the service by exiting with a non-zero status (or to allow it by exiting with a zero

status). It can also look at the $REMOTE_ADDR and **$REMOTE_PORT** environment variables to learn about the requestor when making this decision.

The external command can optionally write a single line to its standard output to be sent to the requestor as an error message when it declines the service.

<directory>
The remaining arguments provide a list of directories. If any directories are specified, then the **git-daemon** process will serve a requested directory only if it is contained in one of these directories. If **--strict-paths** is specified, then the requested directory must match one of these directories exactly.

SERVICES

These services can be globally enabled/disabled using the command-line options of this command. If finer-grained control is desired (e.g. to allow *git archive* to be run against only in a few selected repositories the daemon serves), the per-repository configuration file can be used to enable or disable them.

upload-pack
This serves *git fetch-pack* and *git ls-remote* clients. It is enabled by default, but a repository can disable it by setting **daemon.uploadpack** configuration item to **false**.

upload-archive
This serves *git archive --remote*. It is disabled by default, but a repository can enable it by setting **daemon.uploadarch** configuration item to **true**.

receive-pack
This serves *git send-pack* clients, allowing anonymous push. It is disabled by default, as there is *no* authentication in the protocol (in other words, anybody can push anything into the repository, including removal of refs). This is solely meant for a closed LAN setting where everybody is friendly. This service can be enabled by setting **daemon.receivepack** configuration item to **true**.

EXAMPLES

We assume the following in /etc/services

```
$ grep 9418 /etc/services
git         9418/tcp            # Git Version Control System
```

git daemon as inetd server

To set up *git daemon* as an inetd service that handles any repository within **/pub/foo** or **/pub/bar**, place an entry like the following into **/etc/inetd** all on one line:

```
git stream tcp nowait nobody  /usr/bin/git
    git daemon --inetd --verbose --export-all
    /pub/foo /pub/bar
```

git daemon as inetd server for virtual hosts

To set up *git daemon* as an inetd service that handles repositories for different virtual hosts, **www.example.com** and **www.example.org**, place an entry like the following into **/etc/inetd** all on one line:

```
git stream tcp nowait nobody /usr/bin/git
    git daemon --inetd --verbose --export-all
    --interpolated-path=/pub/%H%D
    /pub/www.example.org/software
    /pub/www.example.com/software
    /software
```

In this example, the root-level directory **/pub** will contain a subdirectory for each virtual host name supported. Further, both hosts advertise repositories simply as **git://www.example.com/software/repo.git**. For pre-1.4.0 clients, a symlink from **/software** into the appropriate default repository could be made as well.

git daemon as regular daemon for virtual hosts

To set up *git daemon* as a regular, non-inetd service that handles repositories for multiple virtual hosts based on their IP addresses, start the daemon like this:

```
git daemon --verbose --export-all
    --interpolated-path=/pub/%IP/%D
    /pub/192.168.1.200/software
    /pub/10.10.220.23/software
```

In this example, the root-level directory **/pub** will contain a subdirectory for each virtual host IP address supported. Repositories can still be accessed by hostname though, assuming they correspond to these IP addresses.

selectively enable/disable services per repository

To enable *git archive --remote* and disable *git fetch* against a repository, have the following in the

configuration file in the repository (that is the file *config* next to **HEAD**, *refs* and *objects*).

```
[daemon]
    uploadpack = false
    uploadarch = true
```

ENVIRONMENT

git daemon will set REMOTE_ADDR to the IP address of the client that connected to it, if the IP address is available. REMOTE_ADDR will be available in the environment of hooks called when services are performed.

GIT

Part of the **git**(1) suite