vms Software

VGIT V0.9-8
A partial git implementation for VSI OpenVMS Alpha and I64

October 2019

1. Introduction

Git (see https://git-scm.com/) is a widely-used distributed version control system for tracking changes in source code throughout the software development lifecycle. VGIT is partial implementation of the git version control system for VSI OpenVMS that supports a growing subset of git functionality, making it possible for the OpenVMS platform and OpenVMS application developers to participate in a heterogeneous application development environment and interact with commonly used software version control services such as GitHub and BitBucket. VGIT is used internally by VMS Software Inc., and it is anticipated that VGIT functionality will be enhanced over time, driven by user requirements.

2. Acknowledgements

VGIT relies upon several Open Source software packages including (but not necessarily limited to) cURL, libgit2, and OpenSSL. VSI Software Inc. would like to thank the developers and maintainers of these packages for their ongoing efforts in developing, supporting, and enhancing this software.

3. What's new in this release

This release of VGIT includes additional functionality from previously releases Beta versions, including a basic implementation of the "stash" command, and functionality to interact with remote repositories via a proxy server.

This release also includes the following changes and updates:

- Changes to address problems that had been observed by users when cloning repositories from the Microsoft Azure DevOps platform.
- It was observed that some operations would fail for certain repositories with the error "`failed to rename lockfile to …`". This error has been resolved.
- The certificate bundle required for VGIT to interact with cloud-based services such as GitHub and BitBucket has been updated to avoid problems with expired certificates. Note that a copy of this file can be obtained at https://raw.githubusercontent.com/bagder/ca-bundle/master/ca-bundle.crt.
- Enhancements to various commands such as `checkout`, `commit`, and `rm` to support additional features and make them more git-like. For example, it is possible to specify wildcards with the `rm` command and the syntax of the VGIT `checkout` command is now more conformant with the equivalent git implementation. Branching is now supported and the `merge` command has been significantly enhanced.

4. Requirements

The kit you are receiving has been compiled, built, and tested using the operating system and compiler versions listed below (or comparable). While it is highly likely that you will have no problems installing and using the kit on systems running higher versions of the products listed here, we cannot say for sure that you will be so lucky if your system is running older versions.

- OpenVMS 8.4-1H1 or higher
- VSI TCP/IP, HPE TCP/IP Services for OpenVMS, or MultiNet TCP/IP stack for network communication

It is assumed that the reader has a good knowledge of OpenVMS and software development in the OpenVMS environment.

Note that OpenSSL 1.1.1b is statically linked with the VGIT image supplied with this kit and it is therefore not strictly necessary to have VSI SSL111 installed on the VSI OpenVMS systems on which you will be using VGIT.

5. Recommended reading

It is recommended that users unfamiliar with git read some of the tutorials and other excellent documentation available on the Internet, including that available at https://git-scm.com/doc.

6. Installing the kit

The kit is provided as an OpenVMS PCSI kit (`VSI-I64VMS-VGIT-V0009-8-1.PCSI` or `VSI-AXPVMS-VGIT-V0009-8-1.PCSI`, depending on platform) that can be installed by a suitably privileged user using the following command:

```
$ PRODUCT INSTALL VGIT
```

The installation will then proceed as follows (output may differ slightly from that shown depending on platform and other factors):

```
Performing product kit validation of signed kits ...

The following product has been selected:
    VSI I64VMS VGIT V0.9-8                 Layered Product

Do you want to continue? [YES]

Configuration phase starting ...

You will be asked to choose options, if any, for each selected
product and for any products that may be installed to satisfy
software dependency requirements.

Configuring VSI I64VMS VGIT V0.9-8

    VMS Software Inc.

* This product does not have any configuration options.

Execution phase starting ...
```

```
The following product will be installed to destination:
    VSI I64VMS VGIT V0.9-8                  DISK$I64SYS:[VMS$COMMON.]

Portion done: 0%...90%...100%

The following product has been installed:
    VSI I64VMS VGIT V0.9-8                  Layered Product

VSI I64VMS VGIT V0.9-8

    Post-installation tasks are required.

    To define logical names required by the VGIT utility, add the
    Following lines to SYS$MANAGER:SYSTARTUP_VMS.COM:

        $ file := SYS$STARTUP:VGIT$STARTUP.COM
        $ if f$search("''file'") .nes. "" then @'file'

    To deassign at system shutdown logical names used by the VGIT
    utility, add the following lines to SYS$MANAGER:SYSHUTDWN.COM:

        $ file := SYS$STARTUP:VGIT$SHUTDOWN.COM
        $ if f$search("''file'") .nes. "" then @'file'

    To allow all users to use the VGIT utility, add the following
    Foreign command definition to SYS$MANAGER:SYLOGIN.COM:

        $ VGIT :== $SYS$SYSTEM:VGIT2.EXE
```

6.1. Post-installation steps

After the installation has successfully completed, follow the instructions displayed at the end of the installation procedure to start using Maven and to ensure that the necessary logical names required in order for users to use the software are defined system-wide at start-up.

Note that the VGIT installation places the file `CA-BUNDLE.CRT` in the `SYS$MANAGER` directory. This file is the certificate bundle that is required by VGIT in order for it to work correctly with cloud-based services such as GitHub and BitBucket. Administrators should take care not to delete this file and if they wish to move the file then the definition of the logical name `GIT$SSL_CERTS` (defined in `VGIT$STARTUP.COM`) should be updated accordingly. It should also be noted that the certificates in `CA-BUNDLE.CRT` do expire and it can be necessary to periodically update the file using the latest version found at https://raw.githubusercontent.com/bagder/ca-bundle/master/ca-bundle.crt.

6.2. Privileges and quotas

Generally speaking there are no special quota or privilege requirements for VGIT, although a reasonably high `BYTLM` and `PGFLQUO` are recommended, particularly if working with large repositories and pushing large updates over network links. The following quotas should be more than adequate for most purposes:

```
Maxjobs:         0  Fillm:       256  Bytlm:       128000
Maxacctjobs:     0  Shrfillm:      0  Pbytlm:           0
Maxdetach:       0  BIOlm:       150  JTquota:       4096
Prclm:          50  DIOlm:       150  WSdef:         4096
```

```
Prio:             4  ASTlm:       300  WSquo:         8192
Queprio:          4  TQElm:       100  WSextent:     16384
CPU:         (none)  Enqlm:      4000  Pgflquo:     256000
```

6.3. Installing in an alternative location

By default the software will be installed in `SYS$SYSDEVICE:[VMS$COMMON]`. If you wish to install the software in an alternative location this can be achieved using the `/DESTINATION` qualifier with the `PRODUCT INSTALL` command to specify the desired location; however it is important to note that an additional manual step will then be required to complete the installation. Specifically, when an alternative destination is specified, the start-up and shutdown procedures (`VGIT$STARTUP.COM` and `VGIT$SHUTDOWN.COM`) will be placed into a subdirectory `[.SYS$STARTUP]` residing under the specified destination directory. If you wish to run these files from your standard `SYS$STARTUP` directory they will need to be copied from the destination subdirectory into your systems `SYS$STARTUP` directory.

7. Getting started

To get started, ensure that the VGIT command and the logical name `GIT$SSL_CERTS` are correctly defined as described in the post-installation tasks. You will then need to run the following command to set up some basic configuration details (substituting your username and email details as appropriate):

```
$ vgit config user -n username -e youremail
```

After this command completes, you should see in your login directory the file `git.config`, and the file should contain the details that you specified above. It is preferable to create this file before doing any operations with remote repositories in order to avoid being prompted for this information when attempting to perform some other operation. Note that the values you specify are not particularly critical and can be overridden if/when necessary; however if you will primarily be interacting with some service such as GitHub or BitBucket then you should specify your username for that service and the associated email address.

If you will need to interact via a proxy server with remote repositories hosted on services such as GitHub and BitBucket now would also be a good time to define details of your proxy server, which can be done using the following command, substituting the correct URL as necessary:

```
$ vgit2 config http.proxy http://myproxy.server:8080
```

VGIT includes some built-in documentation that can be viewed by running the program without specifying a command or specifying an invalid command. Doing this you will get the following output:

```
$ vgit
Usage: $1$DGA100:[SYS0.SYSCOMMON.][SYSEXE]VGIT2.EXE;1 <command> [options] [arguments]
Command summary:
        add [-s] [-v] [-u] <file> ...
        blame [-L <line-range>] [-F] [-M] [-C] [<commit-range>] <path>
        branch [[-d <branch-name>] | [-f] <branch-name> [-t <remote>]]
        checkout [-f] [-b] [branch_name] [-- <file> ...]
        clone [-b <name>] <uri> [<path>]
        commit [-a] [-m <message> | -F <filename>]
        config user [-s global | local] [-n <username>] [-e <email>]
        config http.proxy [-s global | local] <url>
        diff [-s] [-c] [<commit> <commit>] [--] [<path>...]
        fetch [<remote>]
        init [-b] [-q] [-s true | false | group | all | world | umask] [-t <file>]
             [-n (no initial commit)] [<path>]
```

```
        ls-remote <repository>
        merge [-s safe | create | force]
        pull
        push [-t | <refspec>]
        rebase <upstream> <branch>
        remote add <name> <uri>
        remote show
        rm [-b] [-s <stage>] <file>
        show-index
        show-ref
        stash [push | pop | drop | clear]
        status [-b] [-f short | long | porcelain]
        tag -d <tag-name> | [-f] [-m <message>] <tag-name> [<commit> | <object>]
        tag list [-l <number>] [pattern]
        reset [--soft | --mixed | --hard] [<commit>]
        log [<options>] [<revision-range>] [-- <path>...]
        version

        Repositories and your login directory must be on an ODS-5 file systems
        Files must be stream-lf
```

This is a basic summary of the available commands. If you want more details on a specific command, run the program specifying the command in question and supply an invalid option or leave out a required parameter. For example:

```
$ vgit init -h
init: illegal option -- h
Usage: init [-b] [-q] [-s true | false | group | all | world | umask] [-t <file>] [-n]
[<path>]

Options:
        -b              Create a bare repository (has no working tree).
        -q              Quiet (only display errors and warnings).
        -s <value>      Repository sharing (permitted values are "true", "false",
                        "group", "all", "world", and "umask")
        -t <file>       Specify a template directory.
        -n              Do not perform an initial commit.

Arguments:
        <path>          Directory where the repository is to be created (the directory is
                        Created if necessary). If not specified, the current location is
                        used.
```

8. Some points to note

The following list identifies various points that users should be aware of when using VGIT for VSI OpenVMS.

- All files operated on by VGIT must be stream-lf. VGIT will exit with an error if you attempt to add a file to a repository that is not stream-lf. If for any reason the "`add`" command fails, note that nothing will have been added to the repository (the operation will have been rolled back), and you can safely correct the problem (by converting the offending file to stream-lf) and re-run the "`add`" command.
- When adding some number of files VMS Software Inc. would recommend using the "`-v`" (verbose) flag. This will allow you to see which files the add operation is failing on.
- VGIT can be used on ODS-5 formatted file systems only and some VGIT commands are case-sensitive.
- Depending on file system and network performance, initially loading or cloning large repositories (repositories with large numbers of files) may take some time to complete; however subsequent operations such as committing, pushing, and pulling updates is generally fairly efficient, depending on commit/update size. Similarly, the "`status`" command may be a bit slow if there are large numbers of files in your project.

- If you wish to use key-based authentication (as opposed to username/password), you will need to define the logical names `GIT$ID_RSA` and `GIT$ID_RSA_PUB` to map to your private and public key files, respectively (and the public key will need to be registered with the git service that you are using).
- As git supports a variety of authentication methods there can be circumstances when cloning a repository (or performing another operation that requires authentication) where VGIT will by default attempt to use the wrong authentication method. For example, when cloning a repository via ssh VGIT will by default assume public/private key authentication. While for the ssh protocol this is a valid method of authentication, it may not be what you want; you might simply want to authenticate using your username and password. To force this latter behaviour you can define the logical name `GIT$FORCE_CREDTYPE_USERPASS_PLAINTEXT`. You can define this logical name to anything; VGIT checks only for existence.
- Note that it is possible to create a primary repository on shared disk in an OpenVMS cluster, and have team members clone the repository into their private work areas, push changes back to the primary repository, and so on.
- Be wary of logical names matching repository names. Errors may be observed when cloning a repository if there exists a logical name (at any level) that is the same as the repository name. RMS will use the logical name and the associated translation may not be appropriate for the file system operation that is being performed (typically the creation of a directory).