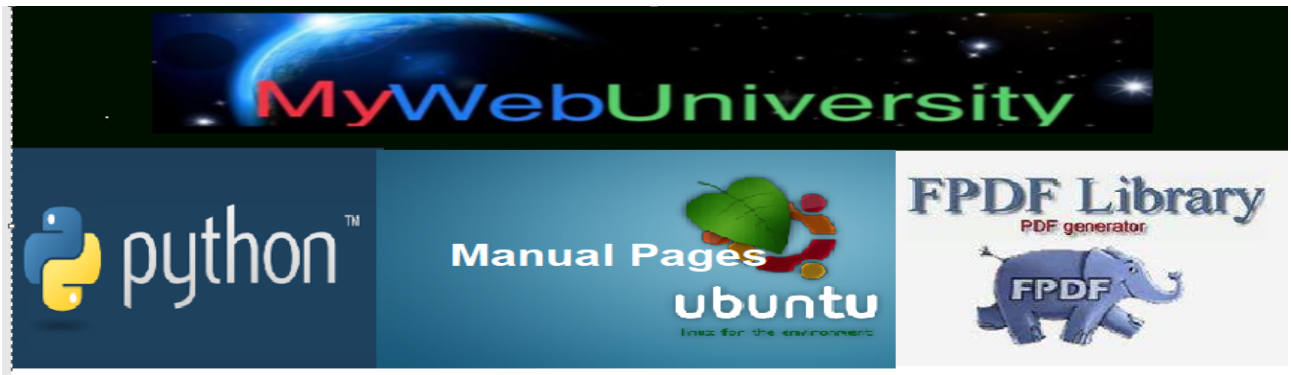

Full credit is given to the above companies including the OS that this PDF file was generated!

Linux Ubuntu 26.04 Manual Pages on command 'ssh-agent.1'

$ man ssh-agent.1

SSH-AGENT(1) General Commands Manual SSH-AGENT(1)

NAME

ssh-agent ? OpenSSH authentication agent

SYNOPSIS

ssh-agent [-c | -s] [-DdTU] [-a bind_address] [-E fingerprint_hash] [-O option]
[-P allowed_providers] [-t life]

ssh-agent [-TU] [-a bind_address] [-E fingerprint_hash] [-O option] [-P allowed_providers]
[-t life] command [arg ...]

ssh-agent [-c | -s] -k

ssh-agent -u

DESCRIPTION

ssh-agent is a program to hold private keys used for public key authentication. Through use of environment variables the agent can be located and automatically used for authentication when logging in to other machines using ssh(1).

The options are as follows:

-a bind_address
Bind the agent to the Unix-domain socket bind_address. The default is to create a socket at a random path matching $HOME/.ssh/agent/s.*.

-c Generate C-shell commands on standard output. This is the default if SHELL looks like it's a csh style of shell.

-D Foreground mode. When this option is specified, ssh-agent will not fork.

-d Debug mode. When this option is specified, ssh-agent will not fork and will write debug information to standard error.

-E fingerprint_hash

Specifies the hash algorithm used when displaying key fingerprints. Valid options are: ?md5? and ?sha256?. The default is ?sha256?.

-k Kill the current agent (given by the SSH_AGENT_PID environment variable).

-O option

Specify an option when starting ssh-agent. The supported options are: allow-remote-pkcs11, no-restrict-websafe and websafe-allow.

The allow-remote-pkcs11 option allows clients of a forwarded ssh-agent to load PKCS#11 or FIDO provider libraries. By default only local clients may perform this operation. Note that signalling that an ssh-agent client is remote is performed by ssh(1), and use of other tools to forward access to the agent socket may circumvent this restriction.

The no-restrict-websafe option instructs ssh-agent to permit signatures using FIDO keys that might be web authentication requests. By default, ssh-agent refuses sig? nature requests for FIDO keys where the key application string does not start with ?ssh:? and when the data to be signed does not appear to be an ssh(1) user authenti? cation request or an ssh-keygen(1) signature. The default behaviour prevents for? warded access to a FIDO key from also implicitly forwarding the ability to authenti? cate to websites.

Alternately the websafe-allow option allows specifying a pattern-list of key appli? cation strings to replace the default application allow-list, for example: ?websafe-allow=ssh:*,example.org,*.example.com?

See PATTERNS in ssh_config(5) for a description of pattern-list syntax.

-P allowed_providers

Specify a pattern-list of acceptable paths for PKCS#11 provider and FIDO authentica? tor middleware shared libraries that may be used with the -S or -s options to ssh-add(1). Libraries that do not match the pattern list will be refused. The de? fault list is ?usr/lib*/*,/usr/local/lib*/*?.

See PATTERNS in ssh_config(5) for a description of pattern-list syntax.

-s Generate Bourne shell commands on standard output. This is the default if SHELL does not look like it's a csh style of shell.

-T Bind the agent socket in a randomised subdirectory of the form $TMPDIR/ssh-XXXXXXXXXX/agent.<ppid>, instead of the default behaviour of using a

randomised name matching $HOME/.ssh/agent/s.*.

-t life

Set a default value for the maximum lifetime of identities added to the agent. The lifetime may be specified in seconds or in a time format specified in sshd_config(5). A lifetime specified for an identity with ssh-add(1) overrides this value. Without this option the default maximum lifetime is forever.

-U Instructs ssh-agent not to clean up stale agent sockets under $HOME/.ssh/agent/.

-u Instructs ssh-agent to only clean up stale agent sockets under $HOME/.ssh/agent/ and then exit immediately. If this option is given twice, ssh-agent will delete stale agent sockets regardless of the host name that created them.

command [arg ...]

If a command (and optional arguments) is given, this is executed as a subprocess of the agent. The agent exits automatically when the command given on the command line terminates.

There are three main ways to get an agent set up. The first is at the start of an X ses? sion, where all other windows or programs are started as children of the ssh-agent program. The agent starts a command under which its environment variables are exported, for example ssh-agent xterm &. When the command terminates, so does the agent.

The second method is used for a login session. When ssh-agent is started, it prints the shell commands required to set its environment variables, which in turn can be evaluated in the calling shell, for example eval `ssh-agent -s`.

In both of these cases, ssh(1) looks at these environment variables and uses them to estab? lish a connection to the agent.

The third way to run ssh-agent is via socket activation from a supervising process, such as systemd. In this mode, the supervising process creates the listening socket and is respon? sible for starting ssh-agent as needed, and also for communicating the location of the socket listener to other programs in the user's session. Socket activation is used when ssh-agent is started with either of the -d or -D flags, no socket listening address speci? fied by the -a flag, and both the LISTEN_FDS and LISTEN_PID environment variables correctly supplied by the supervising process.

The agent initially does not have any private keys. Keys are added using ssh-add(1) or by ssh(1) when AddKeysToAgent is set in ssh_config(5). Multiple identities may be stored in ssh-agent concurrently and ssh(1) will automatically use them if present. ssh-add(1) is

also used to remove keys from ssh-agent and to query the keys that are held in one.

Connections to ssh-agent may be forwarded from further remote hosts using the -A option to ssh(1) (but see the caveats documented therein), avoiding the need for authentication data to be stored on other machines. Authentication passphrases and private keys never go over the network: the connection to the agent is forwarded over SSH remote connections and the result is returned to the requester, allowing the user access to their identities anywhere in the network in a secure fashion.

ssh-agent will delete all keys it has loaded upon receiving SIGUSR1.

ENVIRONMENT

SSH_AGENT_PID When ssh-agent starts, it stores the name of the agent's process ID (PID) in this variable.

SSH_AUTH_SOCK When ssh-agent starts, it creates a Unix-domain socket and stores its path? name in this variable. It is accessible only to the current user, but is easily abused by root or another instance of the same user.

In Debian, ssh-agent is installed with the set-group-id bit set, to prevent ptrace(2) at? tacks retrieving private key material. This has the side-effect of causing the run-time linker to remove certain environment variables which might have security implications for set-id programs, including LD_PRELOAD, LD_LIBRARY_PATH, and TMPDIR. If you need to set any of these environment variables, you will need to do so in the program executed by ssh-agent.

FILES

$HOME/.ssh/agent/s.*

Unix-domain sockets used to contain the connection to the authentication agent. These sockets should only be readable by the owner. The sockets should get automat? ically removed when the agent exits.

SEE ALSO

ssh(1), ssh-add(1), ssh-keygen(1), ssh_config(5), sshd(8)

AUTHORS

OpenSSH is a derivative of the original and free ssh 1.2.12 release by Tatu Ylonen. Aaron Campbell, Bob Beck, Markus Friedl, Niels Provos, Theo de Raadt and Dug Song removed many bugs, re-added newer features and created OpenSSH. Markus Friedl contributed the support for SSH protocol versions 1.5 and 2.0.