PkPass Documentation

Release 2.2.7

Noah Ginsburg, Ryan Adamson

Feb 21, 2023

Contents

CHAPTER 1

Overview

This is a basic password store and password manager for maintaining arbitrary secrets.

The password management solution provides:

- Encryption at Rest
- Password distribution/organization based on definable hierarchies
- Password creation timestamps
- Password history and change logs
- Distributed backup capabilities
- PIV/Smartcard Credential encryption/decryption
- Import and export functionality

Passwords that are created are distributed to recipients by public key encryption. The x509 certificate of the intended recipient is used to create an encrypted copy of the distributed password that is then saved in a password-specific git repository. Multiple encrypted copies of the secret are created, one for each user. End users then check out the git repo and are able to read passwords using their PIV/Smartcard credential to decrypt.

CHAPTER 2

x509 Certificate Repository

PKPass needs a trusted x509 certificate repository, which typically is managed using git. Certificates in this repository should all be signed by Certificate Authorities that can be found in the CABundle file that PKPass is configured to look at. Since this repository should be considered 'trusted', it is typically managed by a smaller trusted set of site administrators. PKPass validates all encryption certificates as they are used to make sure they are signed by a trusted Certificate Authority (CA).

You may also use a local x509 certificate repository that you sync with others using RSYNC, NFS, shared volumes, etc. You can configure the directory that pkpass will use for the certificate repository either on the command line, or through the .pkpassrc file.

The CABundle file to use can also be configured in the .pkpassrc file or on the command line.

Additionally, certificates should be named <username>.cert. For example, the certificate for user 'jason' should be named 'jason.cert' inside this x509 directory.

CHAPTER 3

Password Repository

PKPass also needs a directory to serve as a 'password database'. Like the x509 certificate repository, it is also typically managed with git to provide change control, history, and tracking of changes. Local directories can also be used and shared via rsync, NFS, shared volumes, etc if preferred.

To change the default password repository, you may specify another directory on the command line or in the .pkpassrc file.

3.1 Setup

Pip install is available via:

```
pip install pkpass-olcf
```

Brew install is available via:

```
brew install olcf/tap/pkpass
```

You may clone the pkpass.py tool like this:

```
git clone https://github.com/olcf/pkpass.git
```

If you are using additional PIV/X509 certificate repositories or password repositories, you will need to create local directories for them, or create repositories in a git server that you have access to. Note that while the passwords are safely encrypted and can be distributed without fear of compromise, there may be other information such as system names, account names, and personnel information that you do not want to be publicly available.

3.1.1 RC file

Pkpass has an RC file that can store default values for you so you don't have to write an essay everytime you want to look at or create passwords.

An example file is below

```
certpath: /Users/username/passdb/certs/
keypath: /Users/username/passdb/keys/
cabundle: /Users/username/passdb/cabundles/ca.bundle
pwstore: /Users/username/passdb/passwords/
```

In this case, 'passdb' is the name of the directory in the user's home area that contains x509 certificates, keys (if necessary) and the ca bundle.

The RC file can store any command line argument that is not a true/false value. See Configuration for more details

3.1.2 CA Bundle

You can create a ca bundle by combining all CA Certificates that you trust into one file and moving the file to the cabundle path. Usually the site admins create this CA Bundle for users as part of their certificate management practices. Example

```
cd "${directory_with_ca_certs}"
cat * > ca.bundle
cp ca.bundle "${cabundle_path_in_rc_file}"
```

Additionally, note that most options you can pass on the command line may be passed in through the .pkpassrc file as well. true/false options however (such as –noverify or –nocache), cannot at this time be passed into the command like

3.2 Commands

The Commands can be listed out by passing the help flag to pkpass as seen below

```
usage: pkpass.py [-h] [--config CONFIG] [--version]
                 {card,clip,create,delete,distribute,export,generate,import,info,list,
↪listrecipients,modify,recover,rename,show,update,interpreter}
                 ...

Public Key Password Manager

positional arguments:
  {card,clip,create,delete,distribute,export,generate,import,info,list,listrecipients,
↪modify,recover,rename,show,update,interpreter}
                        sub-commands
    card                List the available cards and which card you have
                        selected
    clip                Copy a password to clipboard
    create              Create a new password entry and encrypt it for
                        yourself
    delete              Delete a password in the repository
    distribute          Distribute existing password entry/ies to another
                        entity [matching uses python fnmatch]
    export              Export passwords that you have access to and encrypt
                        with aes
    generate            Generate a new password entry and encrypt it for
                        yourself
    import              Import passwords that you have saved to a file
    info                Create a new password entry and encrypt it for
                        yourself
    list                List passwords you have access to
```

(continues on next page)

(continued from previous page)

```
    listrecipients      List the recipients that pkpass knows about
    modify              Modify the metadata of a password
    recover             Recover a password that has been distributed using
                        escrow functions
    rename              Rename a password in the repository
    show                Display a password
    update              Change a password value and redistribute to recipients
    interpreter         Interactive mode for pkpass

optional arguments:
  -h, --help            show this help message and exit
  --config CONFIG       Path to a PKPass configuration file. Defaults to
                        '~/.pkpassrc'
  --version             Show the version of PkPass and exit
```

3.2.1 Card

Card lists out available card slots and the currently chosen one

```
usage: pkpass.py card [-h] [--cabundle CABUNDLE] [--certpath CERTPATH]
                      [--color COLOR] [-i IDENTITY] [--no-cache] [-q]
                      [--theme-map THEME_MAP] [-v]

optional arguments:
  -h, --help            show this help message and exit
  --cabundle CABUNDLE   Path to CA certificate bundle file
  --certpath CERTPATH   Path to directory containing public keys. Certificates
                        must end in '.cert'
  --color COLOR         Disable color or not, accepts true/false
  -i IDENTITY, --identity IDENTITY
                        Override identity of user running the program
  --no-cache            if using a connector, pull the certs again
  -q, --quiet           quiet output (show errors only)
  --theme-map THEME_MAP
                        Map of colors to use for colorized output
  -v, --verbose         verbose output (repeat for increased verbosity)
```

3.2.2 Clip

The intent of clip is to copy a password to your clipboard on the unlock event, currently we are aware of a bug with linux systems

```
usage: pkpass.py clip [-h] [--cabundle CABUNDLE] [-c CARD_SLOT]
                      [--certpath CERTPATH] [--color COLOR] [-i IDENTITY]
                      [--keypath KEYPATH] [--no-cache] [--nopassphrase]
                      [--noverify] [--pwstore PWSTORE] [-q] [--stdin]
                      [--theme-map THEME_MAP] [-t TIME] [-v]
                      [pwname]

positional arguments:
  pwname                Name of the password. Ex:
                        passwords/team/infrastructure/root
```

(continues on next page)

(continued from previous page)

```
optional arguments:
  -h, --help            show this help message and exit
  --cabundle CABUNDLE   Path to CA certificate bundle file
  -c CARD_SLOT, --card_slot CARD_SLOT
                        The slot number of the card that should be used
  --certpath CERTPATH   Path to directory containing public keys. Certificates
                        must end in '.cert'
  --color COLOR         Disable color or not, accepts true/false
  -i IDENTITY, --identity IDENTITY
                        Override identity of user running the program
  --keypath KEYPATH     Path to directory containing private keys. Keys must
                        end in '.key'
  --no-cache            if using a connector, pull the certs again
  --nopassphrase, --nopin
                        Do not prompt for a pin/passphrase
  --noverify            Do not verify certificates and signatures
  --pwstore PWSTORE, --srcpwstore PWSTORE
                        Path to the source password store. Defaults to
                        "./passwords"
  -q, --quiet           quiet output (show errors only)
  --stdin               Take all password input from stdin instead of from a
                        user input prompt
  --theme-map THEME_MAP
                        Map of colors to use for colorized output
  -t TIME, --time TIME  Number of seconds to keep password in paste buffer
  -v, --verbose         verbose output (repeat for increased verbosity)
```

3.2.3 Create

Create is used to create a password in the configured password repository

```
usage: pkpass.py create [-h] [--cabundle CABUNDLE] [-c CARD_SLOT]
                        [--certpath CERTPATH] [--color COLOR]
                        [-e ESCROW_USERS] [-i IDENTITY] [--keypath KEYPATH]
                        [-m MIN_ESCROW] [--no-cache] [--noescrow]
                        [--nopassphrase] [--nosign] [--overwrite]
                        [--pwstore PWSTORE] [-q] [--stdin]
                        [--theme-map THEME_MAP] [-v]
                        [pwname]

positional arguments:
  pwname                Name of the password. Ex:
                        passwords/team/infrastructure/root

optional arguments:
  -h, --help            show this help message and exit
  --cabundle CABUNDLE   Path to CA certificate bundle file
  -c CARD_SLOT, --card_slot CARD_SLOT
                        The slot number of the card that should be used
  --certpath CERTPATH   Path to directory containing public keys. Certificates
                        must end in '.cert'
  --color COLOR         Disable color or not, accepts true/false
  -e ESCROW_USERS, --escrow_users ESCROW_USERS
                        Escrow users list is a comma sepearated list of
                        recovery users that each get part of a key
```

(continues on next page)

(continued from previous page)

```
  -i IDENTITY, --identity IDENTITY
                        Override identity of user running the program
  --keypath KEYPATH     Path to directory containing private keys. Keys must
                        end in '.key'
  -m MIN_ESCROW, --min_escrow MIN_ESCROW
                        Minimum number of users required to unlock escrowed
                        password
  --no-cache            if using a connector, pull the certs again
  --noescrow            Do not use escrow functionality, ignore defaults in rc
                        file
  --nopassphrase, --nopin
                        Do not prompt for a pin/passphrase
  --nosign              Do not digitally sign the password information that
                        you are generating
  --overwrite           Overwrite a password that already exists
  --pwstore PWSTORE, --srcpwstore PWSTORE
                        Path to the source password store. Defaults to
                        "./passwords"
  -q, --quiet           quiet output (show errors only)
  --stdin               Take all password input from stdin instead of from a
                        user input prompt
  --theme-map THEME_MAP
                        Map of colors to use for colorized output
  -v, --verbose         verbose output (repeat for increased verbosity)
```

3.2.4 Delete

Delete a password in the repository; pkpass will ask for confirmation. A user could also just remove the file. This is mostly just to allow testing to be a little faster

```
usage: pkpass.py delete [-h] [--cabundle CABUNDLE] [-c CARD_SLOT]
                        [--certpath CERTPATH] [--color COLOR] [-i IDENTITY]
                        [--keypath KEYPATH] [--no-cache] [--overwrite]
                        [--pwstore PWSTORE] [-q] [--stdin]
                        [--theme-map THEME_MAP] [-v]
                        [pwname]

positional arguments:
  pwname                Name of the password. Ex:
                        passwords/team/infrastructure/root

optional arguments:
  -h, --help            show this help message and exit
  --cabundle CABUNDLE   Path to CA certificate bundle file
  -c CARD_SLOT, --card_slot CARD_SLOT
                        The slot number of the card that should be used
  --certpath CERTPATH   Path to directory containing public keys. Certificates
                        must end in '.cert'
  --color COLOR         Disable color or not, accepts true/false
  -i IDENTITY, --identity IDENTITY
                        Override identity of user running the program
  --keypath KEYPATH     Path to directory containing private keys. Keys must
                        end in '.key'
  --no-cache            if using a connector, pull the certs again
  --overwrite           Overwrite a password that already exists
```

(continues on next page)

(continued from previous page)

```
  --pwstore PWSTORE, --srcpwstore PWSTORE
                        Path to the source password store. Defaults to
                        "./passwords"
  -q, --quiet           quiet output (show errors only)
  --stdin               Take all password input from stdin instead of from a
                        user input prompt
  --theme-map THEME_MAP
                        Map of colors to use for colorized output
  -v, --verbose         verbose output (repeat for increased verbosity)
```

3.2.5 Distribute

Distribute takes a pre-existing password in the password repository and grants permission to selected users to be able to unlock it This function resolves filename matching via python's fnmatch module, depending on the string you may need to pass the value through in single quotes

This function will confirm password list is valid even if only one password matches

```
usage: pkpass.py distribute [-h] [--cabundle CABUNDLE] [-c CARD_SLOT]
                            [--certpath CERTPATH] [--color COLOR]
                            [-e ESCROW_USERS] [-g GROUPS] [-i IDENTITY]
                            [--keypath KEYPATH] [-m MIN_ESCROW] [--no-cache]
                            [--noescrow] [--nopassphrase] [--nosign]
                            [--pwstore PWSTORE] [-q] [--stdin]
                            [--theme-map THEME_MAP] [-u USERS] [-v]
                            [pwname]

positional arguments:
  pwname                Name of the password. Ex:
                        passwords/team/infrastructure/root

optional arguments:
  -h, --help            show this help message and exit
  --cabundle CABUNDLE   Path to CA certificate bundle file
  -c CARD_SLOT, --card_slot CARD_SLOT
                        The slot number of the card that should be used
  --certpath CERTPATH   Path to directory containing public keys. Certificates
                        must end in '.cert'
  --color COLOR         Disable color or not, accepts true/false
  -e ESCROW_USERS, --escrow_users ESCROW_USERS
                        Escrow users list is a comma sepearated list of
                        recovery users that each get part of a key
  -g GROUPS, --groups GROUPS
                        Comma seperated list of recipient groups
  -i IDENTITY, --identity IDENTITY
                        Override identity of user running the program
  --keypath KEYPATH     Path to directory containing private keys. Keys must
                        end in '.key'
  -m MIN_ESCROW, --min_escrow MIN_ESCROW
                        Minimum number of users required to unlock escrowed
                        password
  --no-cache            if using a connector, pull the certs again
  --noescrow            Do not use escrow functionality, ignore defaults in rc
                        file
  --nopassphrase, --nopin
```

(continues on next page)

(continued from previous page)

```
                        Do not prompt for a pin/passphrase
  --nosign              Do not digitally sign the password information that
                        you are generating
  --pwstore PWSTORE, --srcpwstore PWSTORE
                        Path to the source password store. Defaults to
                        "./passwords"
  -q, --quiet           quiet output (show errors only)
  --stdin               Take all password input from stdin instead of from a
                        user input prompt
  --theme-map THEME_MAP
                        Map of colors to use for colorized output
  -u USERS, --users USERS
                        Comma seperated list of recipients
  -v, --verbose         verbose output (repeat for increased verbosity)
```

3.2.6 Export

Export allows the current user to migrate all his passwords to one file, this tends to be used in conjunction with import

```
usage: pkpass.py export [-h] [--cabundle CABUNDLE] [-c CARD_SLOT]
                        [--certpath CERTPATH] [--color COLOR]
                        [-i IDENTITY] [--no-cache]
                        [--nocrypto] [--nopassphrase] [-q] [--stdin]
                        [--theme-map THEME_MAP] [-v]
                        [pwfile]

positional arguments:
  pwfile                path to the import/export file

optional arguments:
  -h, --help            show this help message and exit
  --cabundle CABUNDLE   Path to CA certificate bundle file
  -c CARD_SLOT, --card_slot CARD_SLOT
                        The slot number of the card that should be used
  --certpath CERTPATH   Path to directory containing public keys. Certificates
                        must end in '.cert'
  --color COLOR         Disable color or not, accepts true/false
  -i IDENTITY, --identity IDENTITY
                        Override identity of user running the program
  --no-cache            if using a connector, pull the certs again
  --nocrypto            Do not use a password for import/export files
  --nopassphrase, --nopin
                        Do not prompt for a pin/passphrase
  -q, --quiet           quiet output (show errors only)
  --stdin               Take all password input from stdin instead of from a
                        user input prompt
  --theme-map THEME_MAP
                        Map of colors to use for colorized output
  -v, --verbose         verbose output (repeat for increased verbosity)
```

3.2.7 Generate

Generate allows a user to specify a password name and to have the pkpass system generate it based on a regular expression an example rules_map could look like the following

```
usage: pkpass.py generate [-h] [--cabundle CABUNDLE] [-c CARD_SLOT]
                          [--certpath CERTPATH] [--color COLOR]
                          [-e ESCROW_USERS] [-i IDENTITY] [--keypath KEYPATH]
                          [-m MIN_ESCROW] [--no-cache] [--noescrow]
                          [--nopassphrase] [--nosign] [--overwrite]
                          [--pwstore PWSTORE] [-q] [-R RULES]
                          [--rules-map RULES_MAP] [--stdin]
                          [--theme-map THEME_MAP] [-v]
                          [pwname]

positional arguments:
  pwname                Name of the password. Ex:
                        passwords/team/infrastructure/root

optional arguments:
  -h, --help            show this help message and exit
  --cabundle CABUNDLE   Path to CA certificate bundle file
  -c CARD_SLOT, --card_slot CARD_SLOT
                        The slot number of the card that should be used
  --certpath CERTPATH   Path to directory containing public keys. Certificates
                        must end in '.cert'
  --color COLOR         Disable color or not, accepts true/false
  -e ESCROW_USERS, --escrow_users ESCROW_USERS
                        Escrow users list is a comma sepearated list of
                        recovery users that each get part of a key
  -i IDENTITY, --identity IDENTITY
                        Override identity of user running the program
  --keypath KEYPATH     Path to directory containing private keys. Keys must
                        end in '.key'
  -m MIN_ESCROW, --min_escrow MIN_ESCROW
                        Minimum number of users required to unlock escrowed
                        password
  --no-cache            if using a connector, pull the certs again
  --noescrow            Do not use escrow functionality, ignore defaults in rc
                        file
  --nopassphrase, --nopin
                        Do not prompt for a pin/passphrase
  --nosign              Do not digitally sign the password information that
                        you are generating
  --overwrite           Overwrite a password that already exists
  --pwstore PWSTORE, --srcpwstore PWSTORE
                        Path to the source password store. Defaults to
                        "./passwords"
  -q, --quiet           quiet output (show errors only)
  -R RULES, --rules RULES
                        Key of rules to use from provided rules map
  --rules-map RULES_MAP
                        Map of rules used for automated generation of
                        passwords
  --stdin               Take all password input from stdin instead of from a
                        user input prompt
  --theme-map THEME_MAP
                        Map of colors to use for colorized output
  -v, --verbose         verbose output (repeat for increased verbosity)
```

3.2.8 Import

Import allows a user to take an exported password file and import them into a new smart card

```
usage: pkpass.py import [-h] [--cabundle CABUNDLE] [-c CARD_SLOT]
                        [--certpath CERTPATH] [--color COLOR]
                        [-i IDENTITY] [--no-cache]
                        [--nocrypto] [--nopassphrase] [-q] [--stdin]
                        [--theme-map THEME_MAP] [-v]
                        [pwfile]

positional arguments:
  pwfile                path to the import/export file

optional arguments:
  -h, --help            show this help message and exit
  --cabundle CABUNDLE   Path to CA certificate bundle file
  -c CARD_SLOT, --card_slot CARD_SLOT
                        The slot number of the card that should be used
  --certpath CERTPATH   Path to directory containing public keys. Certificates
                        must end in '.cert'
  --color COLOR         Disable color or not, accepts true/false
  -i IDENTITY, --identity IDENTITY
                        Override identity of user running the program
  --no-cache            if using a connector, pull the certs again
  --nocrypto            Do not use a password for import/export files
  --nopassphrase, --nopin
                        Do not prompt for a pin/passphrase
  -q, --quiet           quiet output (show errors only)
  --stdin               Take all password input from stdin instead of from a
                        user input prompt
  --theme-map THEME_MAP
                        Map of colors to use for colorized output
  -v, --verbose         verbose output (repeat for increased verbosity)
```

3.2.9 Info

Info displays metadata to the user about a given password

```
usage: pkpass.py info [-h] [--cabundle CABUNDLE] [--certpath CERTPATH]
                      [--color COLOR] [-i IDENTITY] [--no-cache]
                      [--pwstore PWSTORE] [-q] [--theme-map THEME_MAP] [-v]
                      [pwname]

positional arguments:
  pwname                Name of the password. Ex:
                        passwords/team/infrastructure/root

optional arguments:
  -h, --help            show this help message and exit
  --cabundle CABUNDLE   Path to CA certificate bundle file
  --certpath CERTPATH   Path to directory containing public keys. Certificates
                        must end in '.cert'
  --color COLOR         Disable color or not, accepts true/false
  -i IDENTITY, --identity IDENTITY
                        Override identity of user running the program
```

(continues on next page)

(continued from previous page)

```
  --no-cache            if using a connector, pull the certs again
  --pwstore PWSTORE, --srcpwstore PWSTORE
                        Path to the source password store. Defaults to
                        "./passwords"
  -q, --quiet           quiet output (show errors only)
  --theme-map THEME_MAP
                        Map of colors to use for colorized output
  -v, --verbose         verbose output (repeat for increased verbosity)
```

3.2.10 Interpreter

Creates an interactive session, the default behavior of pkpass if no arguments are passed

```
usage: pkpass.py interpreter [-h] [--cabundle CABUNDLE] [-c CARD_SLOT]
                             [--certpath CERTPATH] [--color COLOR]
                             [--connect CONNECT] [-e ESCROW_USERS] [-g GROUPS]
                             [-i IDENTITY] [--keypath KEYPATH] [-m MIN_ESCROW]
                             [--no-cache] [--pwstore PWSTORE] [-q]
                             [--theme-map THEME_MAP] [-v]

optional arguments:
  -h, --help            show this help message and exit
  --cabundle CABUNDLE   Path to CA certificate bundle file
  -c CARD_SLOT, --card_slot CARD_SLOT
                        The slot number of the card that should be used
  --certpath CERTPATH   Path to directory containing public keys. Certificates
                        must end in '.cert'
  --color COLOR         Disable color or not, accepts true/false
  --connect CONNECT     Connection string for the api to retrieve certs
  -e ESCROW_USERS, --escrow_users ESCROW_USERS
                        Escrow users list is a comma sepearated list of
                        recovery users that each get part of a key
  -g GROUPS, --groups GROUPS
                        Comma seperated list of recipient groups
  -i IDENTITY, --identity IDENTITY
                        Override identity of user running the program
  --keypath KEYPATH     Path to directory containing private keys. Keys must
                        end in '.key'
  -m MIN_ESCROW, --min_escrow MIN_ESCROW
                        Minimum number of users required to unlock escrowed
                        password
  --no-cache            if using a connector, pull the certs again
  --pwstore PWSTORE, --srcpwstore PWSTORE
                        Path to the source password store. Defaults to
                        "./passwords"
  -q, --quiet           quiet output (show errors only)
  --theme-map THEME_MAP
                        Map of colors to use for colorized output
  -v, --verbose         verbose output (repeat for increased verbosity)
```

3.2.11 List

List shows all passwords available to a given user

```
usage: pkpass.py list [-h] [--cabundle CABUNDLE] [--certpath CERTPATH]
                      [--color COLOR] [-f FILTER] [-i IDENTITY] [--no-cache]
                      [--pwstore PWSTORE] [-q] [-r] [--stdin]
                      [--theme-map THEME_MAP] [-v]

optional arguments:
  -h, --help            show this help message and exit
  --cabundle CABUNDLE   Path to CA certificate bundle file
  --certpath CERTPATH   Path to directory containing public keys. Certificates
                        must end in '.cert'
  --color COLOR         Disable color or not, accepts true/false
  -f FILTER, --filter FILTER
                        Reduce output of commands to matching items
  -i IDENTITY, --identity IDENTITY
                        Override identity of user running the program
  --no-cache            if using a connector, pull the certs again
  --pwstore PWSTORE, --srcpwstore PWSTORE
                        Path to the source password store. Defaults to
                        "./passwords"
  -q, --quiet           quiet output (show errors only)
  -r, --recovery        Work with passwords distributed through escrow
                        functionality
  --stdin               Take all password input from stdin instead of from a
                        user input prompt
  --theme-map THEME_MAP
                        Map of colors to use for colorized output
  -v, --verbose         verbose output (repeat for increased verbosity)
```

3.2.12 Listrecipients

List the recipients that pkpass knows about

```
usage: pkpass.py listrecipients [-h] [--cabundle CABUNDLE]
                                [--certpath CERTPATH] [--color COLOR]
                                [-f FILTER] [-i IDENTITY] [--no-cache] [-q]
                                [--stdin] [--theme-map THEME_MAP] [-v]

optional arguments:
  -h, --help            show this help message and exit
  --cabundle CABUNDLE   Path to CA certificate bundle file
  --certpath CERTPATH   Path to directory containing public keys. Certificates
                        must end in '.cert'
  --color COLOR         Disable color or not, accepts true/false
  -f FILTER, --filter FILTER
                        Reduce output of commands to matching items
  -i IDENTITY, --identity IDENTITY
                        Override identity of user running the program
  --no-cache            if using a connector, pull the certs again
  -q, --quiet           quiet output (show errors only)
  --stdin               Take all password input from stdin instead of from a
                        user input prompt
  --theme-map THEME_MAP
                        Map of colors to use for colorized output
  -v, --verbose         verbose output (repeat for increased verbosity)
```

3.2.13 Modify

Modify the metadata of a given password

```
usage: pkpass.py modify [-h] [--cabundle CABUNDLE] [--certpath CERTPATH]
                        [--color COLOR] [-i IDENTITY] [--no-cache]
                        [--pwstore PWSTORE] [-q] [--theme-map THEME_MAP] [-v]
                        [pwname]

positional arguments:
  pwname                Name of the password. Ex:
                        passwords/team/infrastructure/root

optional arguments:
  -h, --help            show this help message and exit
  --cabundle CABUNDLE   Path to CA certificate bundle file
  --certpath CERTPATH   Path to directory containing public keys. Certificates
                        must end in '.cert'
  --color COLOR         Disable color or not, accepts true/false
  -i IDENTITY, --identity IDENTITY
                        Override identity of user running the program
  --no-cache            if using a connector, pull the certs again
  --pwstore PWSTORE, --srcpwstore PWSTORE
                        Path to the source password store. Defaults to
                        "./passwords"
  -q, --quiet           quiet output (show errors only)
  --theme-map THEME_MAP
                        Map of colors to use for colorized output
  -v, --verbose         verbose output (repeat for increased verbosity)
```

3.2.14 Recover

Recover serves the purpose of recovering escrowed passwords in the event no one in the distributed list can properly unlock a password. This requires password owners to have created escrow users. Each necessary escrow user will place his share into the program.

```
usage: pkpass.py recover [-h] [--cabundle CABUNDLE] [--certpath CERTPATH]
                         [--color COLOR] [-e ESCROW_USERS] [-i IDENTITY]
                         [--keypath KEYPATH] [-m MIN_ESCROW] [--no-cache]
                         [--nosign] [--pwstore PWSTORE] [-q]
                         [--theme-map THEME_MAP] [-v]

optional arguments:
  -h, --help            show this help message and exit
  --cabundle CABUNDLE   Path to CA certificate bundle file
  --certpath CERTPATH   Path to directory containing public keys. Certificates
                        must end in '.cert'
  --color COLOR         Disable color or not, accepts true/false
  -e ESCROW_USERS, --escrow_users ESCROW_USERS
                        Escrow users list is a comma sepearated list of
                        recovery users that each get part of a key
  -i IDENTITY, --identity IDENTITY
                        Override identity of user running the program
  --keypath KEYPATH     Path to directory containing private keys. Keys must
                        end in '.key'
  -m MIN_ESCROW, --min_escrow MIN_ESCROW
```

(continues on next page)

(continued from previous page)

```
                        Minimum number of users required to unlock escrowed
                        password
  --no-cache            if using a connector, pull the certs again
  --nosign              Do not digitally sign the password information that
                        you are generating
  --pwstore PWSTORE, --srcpwstore PWSTORE
                        Path to the source password store. Defaults to
                        "./passwords"
  -q, --quiet           quiet output (show errors only)
  --theme-map THEME_MAP
                        Map of colors to use for colorized output
  -v, --verbose         verbose output (repeat for increased verbosity)
```

3.2.15 Rename

This renames a password in the given repository

```
usage: pkpass.py rename [-h] [--cabundle CABUNDLE] [-c CARD_SLOT]
                        [--certpath CERTPATH] [--color COLOR] [-i IDENTITY]
                        [--keypath KEYPATH] [--no-cache] [--nopassphrase]
                        [--overwrite] [--pwstore PWSTORE] [-q] [--stdin]
                        [--theme-map THEME_MAP] [-v]
                        [pwname] [rename]

positional arguments:
  pwname                Name of the password. Ex:
                        passwords/team/infrastructure/root
  rename                New name of the password.

optional arguments:
  -h, --help            show this help message and exit
  --cabundle CABUNDLE   Path to CA certificate bundle file
  -c CARD_SLOT, --card_slot CARD_SLOT
                        The slot number of the card that should be used
  --certpath CERTPATH   Path to directory containing public keys. Certificates
                        must end in '.cert'
  --color COLOR         Disable color or not, accepts true/false
  -i IDENTITY, --identity IDENTITY
                        Override identity of user running the program
  --keypath KEYPATH     Path to directory containing private keys. Keys must
                        end in '.key'
  --no-cache            if using a connector, pull the certs again
  --nopassphrase, --nopin
                        Do not prompt for a pin/passphrase
  --overwrite           Overwrite a password that already exists
  --pwstore PWSTORE, --srcpwstore PWSTORE
                        Path to the source password store. Defaults to
                        "./passwords"
  -q, --quiet           quiet output (show errors only)
  --stdin               Take all password input from stdin instead of from a
                        user input prompt
  --theme-map THEME_MAP
                        Map of colors to use for colorized output
  -v, --verbose         verbose output (repeat for increased verbosity)
```

3.2.16 Show

This unlocks a password and displays it on stdout

```
usage: pkpass.py show [-h] [-a] [-b BEHALF] [--cabundle CABUNDLE]
                      [-c CARD_SLOT] [--certpath CERTPATH] [--color COLOR]
                      [-i IDENTITY] [-I] [--keypath KEYPATH] [--no-cache]
                      [--nopassphrase] [--noverify] [--pwstore PWSTORE] [-q]
                      [-r] [--stdin] [--theme-map THEME_MAP] [-v]
                      [pwname]

positional arguments:
  pwname                Name of the password. Ex:
                    passwords/team/infrastructure/root

optional arguments:
  -h, --help            show this help message and exit
  -a, --all             Show all available password to the given user, if a
                        pwname is supplied filtering will be done case-
                        insensitivey based on the filename
  -b BEHALF, --behalf BEHALF
                        Show passwords for a user using a password as its
                        private key
  --cabundle CABUNDLE   Path to CA certificate bundle file
  -c CARD_SLOT, --card_slot CARD_SLOT
                        The slot number of the card that should be used
  --certpath CERTPATH   Path to directory containing public keys. Certificates
                        must end in '.cert'
  --color COLOR         Disable color or not, accepts true/false
  -i IDENTITY, --identity IDENTITY
                        Override identity of user running the program
  -I, --ignore-decrypt  Ignore decryption errors during show all process
  --keypath KEYPATH     Path to directory containing private keys. Keys must
                        end in '.key'
  --no-cache            if using a connector, pull the certs again
  --nopassphrase, --nopin
                        Do not prompt for a pin/passphrase
  --noverify            Do not verify certificates and signatures
  --pwstore PWSTORE, --srcpwstore PWSTORE
                        Path to the source password store. Defaults to
                        "./passwords"
  -q, --quiet           quiet output (show errors only)
  -r, --recovery        Work with passwords distributed through escrow
                        functionality
  --stdin               Take all password input from stdin instead of from a
                        user input prompt
  --theme-map THEME_MAP
                        Map of colors to use for colorized output
  -v, --verbose         verbose output (repeat for increased verbosity)
```

3.2.17 Update

This changes a password value and redistributes the password to the recipients

```
usage: pkpass.py update [-h] [--cabundle CABUNDLE] [-c CARD_SLOT]
                        [--certpath CERTPATH] [--color COLOR]
```

(continues on next page)

(continued from previous page)

```
                        [-e ESCROW_USERS] [-i IDENTITY] [--keypath KEYPATH]
                        [-m MIN_ESCROW] [--no-cache] [--noescrow]
                        [--nopassphrase] [--nosign] [--overwrite]
                        [--pwstore PWSTORE] [-q] [--stdin]
                        [--theme-map THEME_MAP] [-v]
                        [pwname]

positional arguments:
  pwname                Name of the password. Ex:
                        passwords/team/infrastructure/root

optional arguments:
  -h, --help            show this help message and exit
  --cabundle CABUNDLE   Path to CA certificate bundle file
  -c CARD_SLOT, --card_slot CARD_SLOT
                        The slot number of the card that should be used
  --certpath CERTPATH   Path to directory containing public keys. Certificates
                        must end in '.cert'
  --color COLOR         Disable color or not, accepts true/false
  -e ESCROW_USERS, --escrow_users ESCROW_USERS
                        Escrow users list is a comma sepearated list of
                        recovery users that each get part of a key
  -i IDENTITY, --identity IDENTITY
                        Override identity of user running the program
  --keypath KEYPATH     Path to directory containing private keys. Keys must
                        end in '.key'
  -m MIN_ESCROW, --min_escrow MIN_ESCROW
                        Minimum number of users required to unlock escrowed
                        password
  --no-cache            if using a connector, pull the certs again
  --noescrow            Do not use escrow functionality, ignore defaults in rc
                        file
  --nopassphrase, --nopin
                        Do not prompt for a pin/passphrase
  --nosign              Do not digitally sign the password information that
                        you are generating
  --overwrite           Overwrite a password that already exists
  --pwstore PWSTORE, --srcpwstore PWSTORE
                        Path to the source password store. Defaults to
                        "./passwords"
  -q, --quiet           quiet output (show errors only)
  --stdin               Take all password input from stdin instead of from a
                        user input prompt
  --theme-map THEME_MAP
                        Map of colors to use for colorized output
  -v, --verbose         verbose output (repeat for increased verbosity)
```

3.3 General Usage

Run ./pkpass.py with the '-h' flag for a list of options as well as syntax. Some common usage examples follow:

- Create a new security team root password in the password store:

```
./pkpass.py create security-team/rootpw
```

- Distribute the security team root password to other team members 'foo' and 'bar':

```
./pkpass.py distribute security-team/rootpw -u foo,bar
```

- Distribute the security team passwords to the group secadmins

```
./pkpass.py distribute 'security-team/*' -g secadmins
```

- List the names of all passwords that have been distributed to you:

```
./pkpass.py list
```

- List the names of all escrow passwords that have been distributed to you:

```
./pkpass.py list -r
```

- Show the infrastructure team root password:

```
./pkpass.py show infra-team/rootpw
```

- Show all the passwords that you know:

```
./pkpass.py show -a
```

- Show all the passwords that you know whose filename has rpm (case-insensitive):

```
./pkpass.py show -a rpm
```

- List the names of all passwords that have been distributed to user identity 'foo':

```
./pkpass.py list -i foo
```

- Show the users that pkpass detects certificates for in the certificate repository:

```
./pkpass.py listrecipients
```

3.4 Configuration

3.4.1 Password Repository

Passwords are created on the file system, so any destination may be specified. For passwords that need to be distributed to other users, convention suggests putting these into a hierarchy with the root in 'passwords'. To make the repository as flat as possible, the top level will contain mostly groupings of passwords, with the next level containing the passwords themselves. Examples of groups may include "security-team", "database-users", "passwords/general", etc. It is up to each organization to determine the best hierarchy for storing passwords. The 'list' command and 'showall' commands will crawl the hierarchy starting at the root regardless of structure.

You may distribute passwords to a specified group defined in your pkpassrc file. These groups may be arbitrary

```
databaseadmins: db1, db2,db3
secadmins: admin1,  admin2 ,  admin3
groups: secadmins, databaseadmins
```

you may also specify on the command line which groups to use: `pkpass.py distribute password -g secadmins`

3.4.2 Cert Repository

Certs are read into PkPass and are used in many of the processes. This can be presented to pkpass as a directory structure, repository, or by means of it's `connector` functionality.

3.4.3 CA Bundle

The CA bundle is used to verify valid certs

3.4.4 Arguments

The RC file (location ~/.pkpassrc, ~/.pkpassrc.yaml, or ~/.pkpassrc.yml) can take the majority of PkPass's arguments so that you do not need to pass them through. The only ones that should not be relied upon to work properly are arguments with 'store_true' or 'store_false' attributes. The following arguments should work in a pkpassrc file

```
cabundle
card_slot
certpath
color
connect
escrow_users
groups
identity
keypath
min_escrow
pwstore
rules
rules_map
theme_map
time
users
```

These along with user-defined groups should all work in an RC file.

3.4.5 Special Treatment for Non-piv accounts/credentials

There are some capabilities built into pkpass.py to manage passwords with rsa keys and x509 certificates without using smart card authentication. These keys still need to be signed by a CA in the CA bundle. Create a keypair:

This will create an unsigned keypair. We really want it to create a certificate request in the future

```
openssl req -newkey rsa:4096 -keyout local.key -x509 -out local.cert
```

As long as the private and public keys are in directories that pkpass can find, distribution to those identities works exactly the same. Keys must be named 'username.key'. For user foo, the private key must be named 'foo.key' and reside in the keypath directory.

3.4.6 Behalf of functionality

To utilize the functionality for showing a password on behalf of another user you need to create a password that is the private key of this user. Then when you issue a show command you specify the username with the *-b* flag

Example:

```
pkpass show password_i_dont_have_direct_access_to -b rsa_user
```

the argument *rsa_user* needs to be both the username and the password name for the password that store's this user's rsa key

3.4.7 Populate other data stores

Currently Pkpass can populate puppet-eyaml given appropriate configurations:

It is suggested to have a *~/.eyaml/config.yaml* setup with *pkcs7_public_key:* defined at the highest level of that file.

To completely configure this integration on the pkpass side please add values to your rc file that looks similar to the following

```
populate:
  # puppet_eyaml is the definition for the `type`
  puppet_eyaml:
    # `bin` is the location of the binary for `eyaml`
    bin: /opt/puppetlabs/pdk/share/cache/ruby/2.5.0/bin/eyaml
    # `directory` is the directory of your puppet repo
    directory: ~/git/puppet
    passwords:
      # This level entry (`ops/password`) represents a pkpass password name
      ops/password:
        # This level entry (`data/team/security.yaml`) represents the rest of the␣
↪file path for the heira file
        data/team/security.yaml:
          # The following list represents the keys that need to be replaced in the␣
↪heira file
          - some::server::password
          - some:other::server
```

To populate kubernetes you need a similar block Currently pkpass can only generate a single encrypted value per secret. It places the value stored in pkpass in the map where it's name is matched.

in the following example you will see this, so for *testpass* pkpass will decrypt *testpass* and place the value of that password in *data/password* because in the configuration file the value of *data/password* is *testpass*

Pkpass will then base64 encode all values in the *data* map and dump it as a yaml file in where *output* is defined, in this case */tmp/secrets.yaml*

```
populate:
  kubernetes:
    output: /tmp/secrets.yaml
    passwords:
      testpass:
        - apiVersion: v1
          type: Opaque
          metadata:
            name: test
            namespace: testing
```

(continues on next page)

(continued from previous page)

```
    data:
      password: testpass
      username: someuser
  - apiVersion: v1
    type: Opaque
    metadata:
      name: test
      namespace: testing2
    data:
      password: testpass
      username: someuser
```

It is not recommended to store the kubernetes output file anywhere, since kubernetes secrets are just base64 encoded, they are not secure!

other data endpoints may be requested

3.5 Development and Testing

3.5.1 Testing Scripts

Currently there exists a shell script `./test/pki/generatepki.sh` that will generate certificates for a developer to use for unittests After running this script, you can run tox or the `python -m unittest discover` note that `python -m unittest discover` does not test multiple versions of python like tox does

3.5.2 Plugin Behavior - Connectors

We currently support dropping arbitary connection plugins into `./libpkpass/connectors` the connectors should return certificates, example usage here is if your organization stores certs in a custom web application, or in ldap or the like, you can create a connector to interface with that and feed pkpass certs in this manner

Connectors will be ignored due to the gitignore, I recommend creating a separate repo for that purpose. To use a connector pkpass needs a `connect` argument

```
connect:
  base_directory: /path/to/local/certs # or /tmp
  ConnectorName:
    arbitary_argument1: aa1_value
    aa2: aa2_value
```

This connect argument is a dictionary, the upper level key is the class that python will attempt to import. This class name should also be in a module that is its name in all lowercase.

Example: the class ConnectorName would be in module connectorname

The value of “ConnectorName” in our example above will all be passed to init as a dictionary. this means that “arbitrary_argument1” and “aa2” will both be available for the connector class As you can see the connect argument is a json file, and as such; you may pass multiple connectors in at the same time.

3.6 Software Dependencies

Pkpass has few dependencies. Fernet is a crypto library used to allow automatic symmetric encrypting. Fernet can be installed u

```
pip install cryptography
```

Other dependencies can be found in requirements.txt

Note: All dependencies will be installed if the setup script is run.

3.7 Windows Consideration

There has not been much (if any) testing around the windows ecosystem. Coding has been attempted to comply with portability standards; but compatibility is not guaranteed. If you need it, feel free to submit a PR