```
% Generated by Sphinx.
\def\sphinxdocclass{report}
\documentclass[letterpaper,12pt,openany,oneside]{sphinxmanual}
\usepackage[utf8]{inputenc}
\DeclareUnicodeCharacter{00A0}{\nobreakspace}
\usepackage{cmap}
\usepackage[T1]{fontenc}
\usepackage[english]{babel}
\usepackage{times}
\usepackage[Sonny]{fncychap}
\usepackage{longtable}
\usepackage{sphinx}
\usepackage{multirow}


\addto\captionsenglish{\renewcommand{\figurename}{Fig. }}
\addto\captionsenglish{\renewcommand{\tablename}{Table }}
\floatname{literal-block}{Listing }



\title{CovertFS Documentation}
\date{March 28, 2016}
\release{0.9.2rc1}
\author{Kyle Gorak, Adam Sjoholm, David Hart, Ryne Flores}
\newcommand{\sphinxlogo}{}
\renewcommand{\releasename}{Release}
\makeindex

\makeatletter
\def\PYG@reset{\let\PYG@it=\relax \let\PYG@bf=\relax%
    \let\PYG@ul=\relax \let\PYG@tc=\relax%
    \let\PYG@bc=\relax \let\PYG@ff=\relax}
\def\PYG@tok#1{\csname PYG@tok@#1\endcsname}
\def\PYG@toks#1+{\ifx\relax#1\empty\else%
    \PYG@tok{#1}\expandafter\PYG@toks\fi}
\def\PYG@do#1{\PYG@bc{\PYG@tc{\PYG@ul{%
    \PYG@it{\PYG@bf{\PYG@ff{#1}}}}}}}
\def\PYG#1#2{\PYG@reset\PYG@toks#1+\relax+\PYG@do{#2}}

\expandafter\def\csname PYG@tok@vi\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.73,0.38,0.84}{##1}}}
\expandafter\def\csname PYG@tok@ss\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.32,0.47,0.09}{##1}}}
\expandafter\def\csname PYG@tok@kt\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.56,0.13,0.00}{##1}}}
\expandafter\def\csname PYG@tok@mb\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.13,0.50,0.31}{##1}}}
\expandafter\def\csname PYG@tok@sb\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.25,0.44,0.63}{##1}}}
\expandafter\def\csname PYG@tok@cp\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.00,0.44,0.13}{##1}}}
\expandafter\def\csname PYG@tok@c\endcsname{\let\PYG@it=\textit\def\PYG@tc##1{\textcolor[rgb]{0.25,0.50,0.56}{##1}}}
\expandafter\def\csname PYG@tok@gs\endcsname{\let\PYG@bf=\textbf}
\expandafter\def\csname PYG@tok@k\endcsname{\let\PYG@bf=\textbf\def\PYG@tc##1{\textcolor[rgb]{0.00,0.44,0.13}{##1}}}
\expandafter\def\csname PYG@tok@se\endcsname{\let\PYG@bf=\textbf\def\PYG@tc##1{\textcolor[rgb]{0.25,0.44,0.63}
{##1}}}
\expandafter\def\csname PYG@tok@nb\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.00,0.44,0.13}{##1}}}
\expandafter\def\csname PYG@tok@m\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.13,0.50,0.31}{##1}}}
\expandafter\def\csname PYG@tok@err\endcsname{\def\PYG@bc##1{\setlength{\fboxsep}{0pt}\fcolorbox[rgb]
{1.00,0.00,0.00}{1,1,1}{\strut ##1}}}
\expandafter\def\csname PYG@tok@ow\endcsname{\let\PYG@bf=\textbf\def\PYG@tc##1{\textcolor[rgb]{0.00,0.44,0.13}
{##1}}}
\expandafter\def\csname PYG@tok@s2\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.25,0.44,0.63}{##1}}}
\expandafter\def\csname PYG@tok@nn\endcsname{\let\PYG@bf=\textbf\def\PYG@tc##1{\textcolor[rgb]{0.05,0.52,0.71}
```

```
{##1}}}
\expandafter\def\csname PYG@tok@cm\endcsname{\let\PYG@it=\textit\def\PYG@tc##1{\textcolor[rgb]{0.25,0.50,0.56}{##1}}}
\expandafter\def\csname PYG@tok@gh\endcsname{\let\PYG@bf=\textbf\def\PYG@tc##1{\textcolor[rgb]{0.00,0.00,0.50}
{##1}}}
\expandafter\def\csname PYG@tok@mh\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.13,0.50,0.31}{##1}}}
\expandafter\def\csname PYG@tok@mo\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.13,0.50,0.31}{##1}}}
\expandafter\def\csname PYG@tok@si\endcsname{\let\PYG@it=\textit\def\PYG@tc##1{\textcolor[rgb]{0.44,0.63,0.82}{##1}}}
\expandafter\def\csname PYG@tok@gr\endcsname{\def\PYG@tc##1{\textcolor[rgb]{1.00,0.00,0.00}{##1}}}
\expandafter\def\csname PYG@tok@ni\endcsname{\let\PYG@bf=\textbf\def\PYG@tc##1{\textcolor[rgb]{0.84,0.33,0.22}
{##1}}}
\expandafter\def\csname PYG@tok@nc\endcsname{\let\PYG@bf=\textbf\def\PYG@tc##1{\textcolor[rgb]{0.05,0.52,0.71}
{##1}}}
\expandafter\def\csname PYG@tok@kr\endcsname{\let\PYG@bf=\textbf\def\PYG@tc##1{\textcolor[rgb]{0.00,0.44,0.13}
{##1}}}
\expandafter\def\csname PYG@tok@kc\endcsname{\let\PYG@bf=\textbf\def\PYG@tc##1{\textcolor[rgb]{0.00,0.44,0.13}
{##1}}}
\expandafter\def\csname PYG@tok@kd\endcsname{\let\PYG@bf=\textbf\def\PYG@tc##1{\textcolor[rgb]{0.00,0.44,0.13}
{##1}}}
\expandafter\def\csname PYG@tok@gu\endcsname{\let\PYG@bf=\textbf\def\PYG@tc##1{\textcolor[rgb]{0.50,0.00,0.50}
{##1}}}
\expandafter\def\csname PYG@tok@gd\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.63,0.00,0.00}{##1}}}
\expandafter\def\csname PYG@tok@gi\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.00,0.63,0.00}{##1}}}
\expandafter\def\csname PYG@tok@mf\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.13,0.50,0.31}{##1}}}
\expandafter\def\csname PYG@tok@go\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.20,0.20,0.20}{##1}}}
\expandafter\def\csname PYG@tok@kp\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.00,0.44,0.13}{##1}}}
\expandafter\def\csname PYG@tok@sx\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.78,0.36,0.04}{##1}}}
\expandafter\def\csname PYG@tok@nf\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.02,0.16,0.49}{##1}}}
\expandafter\def\csname PYG@tok@nv\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.73,0.38,0.84}{##1}}}
\expandafter\def\csname PYG@tok@cs\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.25,0.50,0.56}{##1}}\def
\PYG@bc##1{\setlength{\fboxsep}{0pt}\colorbox[rgb]{1.00,0.94,0.94}{\strut ##1}}}
\expandafter\def\csname PYG@tok@nd\endcsname{\let\PYG@bf=\textbf\def\PYG@tc##1{\textcolor[rgb]{0.33,0.33,0.33}
{##1}}}
\expandafter\def\csname PYG@tok@s1\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.25,0.44,0.63}{##1}}}
\expandafter\def\csname PYG@tok@c1\endcsname{\let\PYG@it=\textit\def\PYG@tc##1{\textcolor[rgb]{0.25,0.50,0.56}{##1}}}
\expandafter\def\csname PYG@tok@vc\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.73,0.38,0.84}{##1}}}
\expandafter\def\csname PYG@tok@bp\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.00,0.44,0.13}{##1}}}
\expandafter\def\csname PYG@tok@sc\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.25,0.44,0.63}{##1}}}
\expandafter\def\csname PYG@tok@sd\endcsname{\let\PYG@it=\textit\def\PYG@tc##1{\textcolor[rgb]{0.25,0.44,0.63}{##1}}}
\expandafter\def\csname PYG@tok@s\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.25,0.44,0.63}{##1}}}
\expandafter\def\csname PYG@tok@mi\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.13,0.50,0.31}{##1}}}
\expandafter\def\csname PYG@tok@no\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.38,0.68,0.84}{##1}}}
\expandafter\def\csname PYG@tok@nl\endcsname{\let\PYG@bf=\textbf\def\PYG@tc##1{\textcolor[rgb]{0.00,0.13,0.44}
{##1}}}
\expandafter\def\csname PYG@tok@gt\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.00,0.27,0.87}{##1}}}
\expandafter\def\csname PYG@tok@sr\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.14,0.33,0.53}{##1}}}
\expandafter\def\csname PYG@tok@o\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.40,0.40,0.40}{##1}}}
\expandafter\def\csname PYG@tok@nt\endcsname{\let\PYG@bf=\textbf\def\PYG@tc##1{\textcolor[rgb]{0.02,0.16,0.45}
{##1}}}
\expandafter\def\csname PYG@tok@il\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.13,0.50,0.31}{##1}}}
\expandafter\def\csname PYG@tok@na\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.25,0.44,0.63}{##1}}}
\expandafter\def\csname PYG@tok@w\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.73,0.73,0.73}{##1}}}
\expandafter\def\csname PYG@tok@sh\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.25,0.44,0.63}{##1}}}
\expandafter\def\csname PYG@tok@kn\endcsname{\let\PYG@bf=\textbf\def\PYG@tc##1{\textcolor[rgb]{0.00,0.44,0.13}
{##1}}}
\expandafter\def\csname PYG@tok@ne\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.00,0.44,0.13}{##1}}}
\expandafter\def\csname PYG@tok@gp\endcsname{\let\PYG@bf=\textbf\def\PYG@tc##1{\textcolor[rgb]{0.78,0.36,0.04}
{##1}}}
\expandafter\def\csname PYG@tok@ge\endcsname{\let\PYG@it=\textit}
\expandafter\def\csname PYG@tok@vg\endcsname{\def\PYG@tc##1{\textcolor[rgb]{0.73,0.38,0.84}{##1}}}
```

\def\PYGZbs{\char`\\}
\def\PYGZus{\char`_}
\def\PYGZob{\char`\{}
\def\PYGZcb{\char`\}}
\def\PYGZca{\char`\^}
\def\PYGZam{\char`\&}
\def\PYGZlt{\char`\<}
\def\PYGZgt{\char`\>}
\def\PYGZsh{\char`\#}
\def\PYGZpc{\char`\%}
\def\PYGZdl{\char`\$}
\def\PYGZhy{\char`\-}
\def\PYGZsq{\char`\'}
\def\PYGZdq{\char`\"}
\def\PYGZti{\char`\~}
% for compatibility with earlier versions
\def\PYGZat{@}
\def\PYGZlb{[}
\def\PYGZrb{]}
\makeatother

\renewcommand\PYGZsq{\textquotesingle}

\begin{document}

\maketitle
\tableofcontents
\phantomsection\label{index::doc}

\chapter{Source Documentation}
\label{index:source-documentation}

\section{Command line main module}
\label{main::doc}\label{main:command-line-main-module}\label{main:module-main}\index{main (module)}
\emph{covertFS} is a command line based program created using the \emph{cmd} module.
The \emph{main.py} file contains all commands available and additional helper
functions.
\index{Console (class in main)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console}\pysiglinewithargsret{\strong{class }\code{main.}\bfcode{Console}}{\emph{online
_file_store}, \emph{steg_class}, \emph{encrypt_class}, \emph{mountpoint}, \emph{url}, \emph{proxy},
\emph{cmdLoopUsed}, \emph{dbg=False}}{}
Bases: \code{cmd.Cmd}, \code{object}
\index{add_file_to_fs() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.add_file_to_fs}\pysiglinewithargsret{\bfcode{add_file_to_fs}}{\emph{fspath},
\emph{contents}}{}
Helper function to add a file to the fs.

\end{fulllineitems}

\index{background_upload() (main.Console method)}

\begin{fulllineitems}

\phantomsection\label{main:main.Console.background_upload}\pysiglinewithargsret{\bfcode{background_upload}}{}{}
\end{fulllineitems}

\index{completedefault() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.completedefault}\pysiglinewithargsret{\bfcode{completedefault}}{\emph{text},
\emph{line}, \emph{begidx}, \emph{endidx}}{}
Allow Tab autocompletion of file names.

\end{fulllineitems}

\index{default() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.default}\pysiglinewithargsret{\bfcode{default}}{\emph{line}}{}
Called on an input line when the command prefix is not recognized.
In that case we execute the line as Python code.

\end{fulllineitems}

\index{do_EOF() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.do_EOF}\pysiglinewithargsret{\bfcode{do_EOF}}{\emph{args}}{}
Exit on system end of file character

\end{fulllineitems}

\index{do_cat() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.do_cat}\pysiglinewithargsret{\bfcode{do_cat}}{\emph{args}}{}
View the contents of a file in the file system.

Use: cat {[}covert path{]}

\end{fulllineitems}

\index{do_cd() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.do_cd}\pysiglinewithargsret{\bfcode{do_cd}}{\emph{args}}{}
Change directory to specified {[}path{]}

Use: cd {[}path{]}*

\end{fulllineitems}

\index{do_decodemsg() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.do_decodemsg}\pysiglinewithargsret{\bfcode{do_decodemsg}}{\emph{in_url}}{}
Decode the message in an image.

Returns the message in plain text.

decodeimage {[}download url{]}

\end{fulllineitems}

\index{do_download() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.do_download}\pysiglinewithargsret{\bfcode{do_download}}{\emph{args}}{}
Download a covert file to the local file system.

Use: download {[}covert path{]} {[}local path{]}

\end{fulllineitems}

\index{do_encodemsg() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.do_encodemsg}\pysiglinewithargsret{\bfcode{do_encodemsg}}{\emph{message}}{}
Encode a message and upload to social media.

Returns the url.

Use: encodemsg {[}message{]}

\end{fulllineitems}

\index{do_exit() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.do_exit}\pysiglinewithargsret{\bfcode{do_exit}}{\emph{args}}{}
Exits from the console

\end{fulllineitems}

\index{do_help() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.do_help}\pysiglinewithargsret{\bfcode{do_help}}{\emph{args}}{}~\begin{description}
\item[{Get help on commands}] \leavevmode
`help' or `?' with no arguments prints the list of commands
`help \textless{}command\textgreater{}' or `? \textless{}command\textgreater{}' gives help on \textless{}command\textgreater{}

\end{description}

\end{fulllineitems}

\index{do_hist() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.do_hist}\pysiglinewithargsret{\bfcode{do_hist}}{\emph{args}}{}
Print a list of commands that have been entered

\end{fulllineitems}

\index{do_loadfs() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.do_loadfs}\pysiglinewithargsret{\bfcode{do_loadfs}}{\emph{url}}{}
Load a covert file system.

Use: loadfs {[}url{]}

\end{fulllineitems}

\index{do_ls() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.do_ls}\pysiglinewithargsret{\bfcode{do_ls}}{\emph{args}}{}
List items in directory

Use: ls {[}path{]}*

\end{fulllineitems}

\index{do_mkdir() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.do_mkdir}\pysiglinewithargsret{\bfcode{do_mkdir}}{\emph{args}}{}
Make a folder at the given path.

Use: mkdir {[}path{]}

\end{fulllineitems}

\index{do_mkfile() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.do_mkfile}\pysiglinewithargsret{\bfcode{do_mkfile}}{\emph{args}}{}
Create a text file with a message to the file system.

Use: mkfile {[}path{]} {[}message{]}

\end{fulllineitems}

\index{do_mount() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.do_mount}\pysiglinewithargsret{\bfcode{do_mount}}{\emph{args}}{}
\end{fulllineitems}

\index{do_newfs() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.do_newfs}\pysiglinewithargsret{\bfcode{do_newfs}}{\emph{args}}{}
Create a covert file system, return the URL of the old fs.

Use: newfs

\end{fulllineitems}

\index{do_noproxy() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.do_noproxy}\pysiglinewithargsret{\bfcode{do_noproxy}}{\emph{args}}{}
Turns off the built-in proxy.

Use: noproxy

\end{fulllineitems}

\index{do_proxy() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.do_proxy}\pysiglinewithargsret{\bfcode{do_proxy}}{\emph{args}}{}
Turns on the built-in proxy.

Use: proxy

\end{fulllineitems}

\index{do_rm() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.do_rm}\pysiglinewithargsret{\bfcode{do_rm}}{\emph{args}}{}
Remove a file from the covert file system.

Use: rm {[}path{]}*

\end{fulllineitems}

\index{do_rmdir() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.do_rmdir}\pysiglinewithargsret{\bfcode{do_rmdir}}{\emph{args}}{}
Remove a folder in the current directory.

Use: rmdir {[}path{]}

\end{fulllineitems}

\index{do_shell() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.do_shell}\pysiglinewithargsret{\bfcode{do_shell}}{\emph{args}}{}
Pass command to a system shell when line begins with `!'

\end{fulllineitems}

\index{do_upload() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.do_upload}\pysiglinewithargsret{\bfcode{do_upload}}{\emph{args}}{}
Upload a local file to the covert file system.

Use: upload {[}local path{]} {[}covert path{]}

\end{fulllineitems}

\index{do_uploadfs() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.do_uploadfs}\pysiglinewithargsret{\bfcode{do_uploadfs}}{\emph{args}}{}
Upload covert fileSystem to the web

\end{fulllineitems}

\index{down_and_set_file() (main.Console method)}

\begin{fulllineitems}

\phantomsection\label{main:main.Console.down_and_set_file}\pysiglinewithargsret{\bfcode{down_and_set_file}}
{\emph{filename}}{}
Download a file. Put it in the filesystem.

\end{fulllineitems}

\index{download_file() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.download_file}\pysiglinewithargsret{\bfcode{download_file}}{\emph{url}}{}
\end{fulllineitems}

\index{emptyline() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.emptyline}\pysiglinewithargsret{\bfcode{emptyline}}{}{}
Do nothing on empty input line

\end{fulllineitems}

\index{init_factory() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.init_factory}\pysiglinewithargsret{\bfcode{init_factory}}{}{}
\end{fulllineitems}

\index{loadfs() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.loadfs}\pysiglinewithargsret{\bfcode{loadfs}}{}{}
\end{fulllineitems}

\index{open_window() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.open_window}\pysiglinewithargsret{\bfcode{open_window}}{}{}
\end{fulllineitems}

\index{postcmd() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.postcmd}\pysiglinewithargsret{\bfcode{postcmd}}{\emph{stop}, \emph{line}}{}
If you want to stop the console, return something that evaluates to true.
If you want to do some post command processing, do it here.

\end{fulllineitems}

\index{postloop() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.postloop}\pysiglinewithargsret{\bfcode{postloop}}{}{}
Take care of any unfinished business.
Despite the claims in the Cmd documentaion,
Cmd.postloop() is not a stub.

\end{fulllineitems}

\index{precmd() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.precmd}\pysiglinewithargsret{\bfcode{precmd}}{\emph{line}}{}
This method is called after the line has been input but before
it has been interpreted. If you want to modifdy the input line
before execution (for example, variable substitution) do it here.

\end{fulllineitems}

\index{preloop() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.preloop}\pysiglinewithargsret{\bfcode{preloop}}{}{}
Initialization before prompting user for commands.
Despite the claims in the Cmd documentaion,
Cmd.preloop() is not a stub.

\end{fulllineitems}

\index{san_file() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.san_file}\pysiglinewithargsret{\bfcode{san_file}}{\emph{file_contents}}{}
Sanitize file before 1)viewing contents or 2)putting on host OS

\end{fulllineitems}

\index{upload_file() (main.Console method)}

\begin{fulllineitems}
\phantomsection\label{main:main.Console.upload_file}\pysiglinewithargsret{\bfcode{upload_file}}{\emph{contents}}{}
Helper function to upload file, return the download url.

\end{fulllineitems}

\end{fulllineitems}

\index{proxy_parser() (in module main)}

\begin{fulllineitems}
\phantomsection\label{main:main.proxy_parser}\pysiglinewithargsret{\code{main.}\bfcode{proxy_parser}}
{\emph{proxyString=None}}{}
\end{fulllineitems}

\index{proxy_test() (in module main)}

\begin{fulllineitems}
\phantomsection\label{main:main.proxy_test}\pysiglinewithargsret{\code{main.}\bfcode{proxy_test}}{\emph{proxyL}}{}
\end{fulllineitems}

\section{File_System package}
\label{File_System:file-system-package}\label{File_System::doc}

\subsection{Submodules}
\label{File_System:submodules}

\subsubsection{covertfs module}

\label{File_System:module-File_System.covertfs}\label{File_System:covertfs-module}\index{File_System.covertfs (module)}
The \emph{covertfs} module extends the \emph{pyfilesystem} package, using the \emph{MemoryFS}
file system.
The MemoryFS file system stores all directory and file info in main memory,
to allow for instantaneous file access as well as to avoid writing any FS
information to disk. This allows for plausible deniability. All
filesystem-necessary commands (ls, cd, mkdir, rm etc) are extended in this
module.

The covertfs module includes CovertFile, a subclass of MemoryFile, CovertEntry,
a subclass of MemoryEntry, and CovertFS, a subclass of MemoryFS. CovertFS uses
CovertFile as the file factory, and CovertEntry as the entry factory.
\index{CovertEntry (class in File_System.covertfs)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.covertfs.CovertEntry}\pysiglinewithargsret{\strong{class }\code{File
_System.covertfs.}\bfcode{CovertEntry}}{\emph{type}, \emph{name}, \emph{contents=None}}{}
Bases: \code{File_System.memoryfs.DirEntry}

\end{fulllineitems}

\index{CovertFS (class in File_System.covertfs)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.covertfs.CovertFS}\pysigline{\strong{class }\code{File_System.covertfs.}
\bfcode{CovertFS}}
Bases: \code{File_System.memoryfs.MemoryFS}
\index{addfile() (File_System.covertfs.CovertFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.covertfs.CovertFS.addfile}\pysiglinewithargsret{\bfcode{addfile}}{\emph{path},
\emph{contents}}{}
Add a file, with given contents, to given path.
Error if path is a file, directory, or if parent directory
is not present.

\end{fulllineitems}

\index{cd() (File_System.covertfs.CovertFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.covertfs.CovertFS.cd}\pysiglinewithargsret{\bfcode{cd}}{\emph{path='/'}}{}
Changes current directory. Superclass has no concept of
current directory (all calls are made from root dir), so
this method is purely local.
Error if given path does not exist, or is a file.

\end{fulllineitems}

\index{check_parent_dir() (File_System.covertfs.CovertFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.covertfs.CovertFS.check_parent_dir}\pysiglinewithargsret{\bfcode{check
_parent_dir}}{\emph{path}}{}
Checks to ensure parent directory is present before
attempting to add a file to it.

\end{fulllineitems}

\index{loadfs() (File_System.covertfs.CovertFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.covertfs.CovertFS.loadfs}\pysiglinewithargsret{\bfcode{loadfs}}
{\emph{fsstring}}{}
Iterates through a string that represents the filesystem
(either pulled from online, or given as a test string),
makes necessary directories, and creates necessary files
(empty for now) that are then loaded by main.py.

\end{fulllineitems}

\index{ls() (File_System.covertfs.CovertFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.covertfs.CovertFS.ls}\pysiglinewithargsret{\bfcode{ls}}{\emph{path=None}}{}
Returns a list of the files and directories in the given
path, or the current directory if no path is given.
Error if given path does not exist, or is a file.

\end{fulllineitems}

\index{mkdir() (File_System.covertfs.CovertFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.covertfs.CovertFS.mkdir}\pysiglinewithargsret{\bfcode{mkdir}}{\emph{path}}{}
Makes a new directory at given path.
Error if path is a directory already, or a file.

\end{fulllineitems}

\index{rm() (File_System.covertfs.CovertFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.covertfs.CovertFS.rm}\pysiglinewithargsret{\bfcode{rm}}{\emph{path}}{}
Removes file at given path.
Error if no such file.

\end{fulllineitems}

\index{rmdir() (File_System.covertfs.CovertFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.covertfs.CovertFS.rmdir}\pysiglinewithargsret{\bfcode{rmdir}}
{\emph{path=None}, \emph{force=False}}{}
Removes empty directory at given path, or non-empty
directory if force option is given.
Error if path is not a directory.

\end{fulllineitems}

\index{sanitize_path() (File_System.covertfs.CovertFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.covertfs.CovertFS.sanitize_path}\pysiglinewithargsret{\bfcode{sanitize_path}}
{\emph{path=None}}{}
This method takes a user-input path and makes it
method-readable, by adding the current path to the
front (if the desired path doesn't start with /) or

returning the root path if no path is given.

\end{fulllineitems}

\index{save() (File_System.covertfs.CovertFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.covertfs.CovertFS.save}\pysiglinewithargsret{\bfcode{save}}{}{}
Turns the entire filesystem into a string to be uploaded.
Returns that string.

\end{fulllineitems}

\index{setcreate() (File_System.covertfs.CovertFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.covertfs.CovertFS.setcreate}\pysiglinewithargsret{\bfcode{setcreate}}
{\emph{path}, \emph{timeIn=None}}{}
Sets the creation time of a file or directory. Used when the filesystem is loaded from online repository

\end{fulllineitems}

\index{settimes() (File_System.covertfs.CovertFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.covertfs.CovertFS.settimes}\pysiglinewithargsret{\bfcode{settimes}}
{\emph{path}, \emph{accessed_time=None}, \emph{modified_time=None}}{}
Sets accessed and modified times of a file or directory after file operations take place

\end{fulllineitems}

\end{fulllineitems}

\index{CovertFile (class in File_System.covertfs)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.covertfs.CovertFile}\pysiglinewithargsret{\strong{class }\code{File
_System.covertfs.}\bfcode{CovertFile}}{\emph{path}, \emph{memory_fs}, \emph{mem_file}, \emph{mode}, \emph{lock}}{}
Bases: \code{File_System.memoryfs.MemoryFile}

\end{fulllineitems}

\subsubsection{memfuse module}
\label{File_System:memfuse-module}\label{File_System:module-File_System.memfuse}\index{File_System.memfuse
(module)}
The \emph{memfuse} module extends the \emph{pyfuse} package, specifically the
\emph{Operations} module.
`Operations' is linked to the Unix FUSE package with python ctypes, to make the
file system available to the user, without making system calls. The memfuse
module extends all necessary file system operations in a way that is accessible
by FUSE, effectively linking the CovertFS class to the FUSE package. When
mounted (using FUSE) onto the native Linux file system, memfuse provides access
to all files and directories in the CovertFS file system.
\index{MemFS (class in File_System.memfuse)}

\begin{fulllineitems}

\phantomsection\label{File_System:File_System.memfuse.MemFS}\pysiglinewithargsret{\strong{class }\code{File
_System.memfuse.}\bfcode{MemFS}}{\emph{memoryfs}}{}
Bases: \code{File_System.fuse.LoggingMixIn}, \code{File_System.fuse.Operations}

Subclass of operations. Links all necessary filesystem operations to CovertFS
\index{create() (File_System.memfuse.MemFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.memfuse.MemFS.create}\pysiglinewithargsret{\bfcode{create}}{\emph{path},
\emph{mode}}{}
Makes a new file in mounted filesystem

\end{fulllineitems}

\index{flush() (File_System.memfuse.MemFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.memfuse.MemFS.flush}\pysiglinewithargsret{\bfcode{flush}}{\emph{path},
\emph{fh}}{}
Writes all attributes and extra attributes of a file after it is done being accessed

\end{fulllineitems}

\index{fsync() (File_System.memfuse.MemFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.memfuse.MemFS.fsync}\pysiglinewithargsret{\bfcode{fsync}}{\emph{path},
\emph{fdatasync}, \emph{fh}}{}
Same as flush

\end{fulllineitems}

\index{getattr() (File_System.memfuse.MemFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.memfuse.MemFS.getattr}\pysiglinewithargsret{\bfcode{getattr}}{\emph{path},
\emph{fh=None}}{}
Function used extensively by the OS package FUSE for interaction with system calls

\end{fulllineitems}

\index{getxattr() (File_System.memfuse.MemFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.memfuse.MemFS.getxattr}\pysiglinewithargsret{\bfcode{getxattr}}
{\emph{path}, \emph{name}, \emph{position=0}}{}
Returns an additional attribute of a file (such as SELinux information)

\end{fulllineitems}

\index{listxattr() (File_System.memfuse.MemFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.memfuse.MemFS.listxattr}\pysiglinewithargsret{\bfcode{listxattr}}
{\emph{path}}{}
Lists extra attributes of a file

\end{fulllineitems}

\index{mkdir() (File_System.memfuse.MemFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.memfuse.MemFS.mkdir}\pysiglinewithargsret{\bfcode{mkdir}}{\emph{path}, \emph{mode}}{}
Make a directory in mounted filesystem

\end{fulllineitems}

\index{open() (File_System.memfuse.MemFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.memfuse.MemFS.open}\pysiglinewithargsret{\bfcode{open}}{\emph{path}, \emph{flags}}{}
Opens the file at a given path in mounted filesystem

\end{fulllineitems}

\index{read() (File_System.memfuse.MemFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.memfuse.MemFS.read}\pysiglinewithargsret{\bfcode{read}}{\emph{path}, \emph{size}, \emph{offset}, \emph{fh}}{}
Reads the contents of a file and returns the requested number of bytes

\end{fulllineitems}

\index{readdir() (File_System.memfuse.MemFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.memfuse.MemFS.readdir}\pysiglinewithargsret{\bfcode{readdir}}{\emph{path}, \emph{fh}}{}
Same as listing the contents of a directory

\end{fulllineitems}

\index{removexattr() (File_System.memfuse.MemFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.memfuse.MemFS.removexattr}\pysiglinewithargsret{\bfcode{removexattr}}{\emph{path}, \emph{name}}{}
Removes the extra attribute from file

\end{fulllineitems}

\index{rename() (File_System.memfuse.MemFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.memfuse.MemFS.rename}\pysiglinewithargsret{\bfcode{rename}}{\emph{old}, \emph{new}}{}
Rename a file or directory in a mounted filesystem

\end{fulllineitems}

\index{rmdir() (File_System.memfuse.MemFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.memfuse.MemFS.rmdir}\pysiglinewithargsret{\bfcode{rmdir}}{\emph{path}}{}
Removes a directory from mounted filesystem

\end{fulllineitems}

\index{setxattr() (File_System.memfuse.MemFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.memfuse.MemFS.setxattr}\pysiglinewithargsret{\bfcode{setxattr}}
{\emph{path}, \emph{name}, \emph{value}, \emph{options}, \emph{position=0}}{}
Sets the extra attribute to given value

\end{fulllineitems}

\index{truncate() (File_System.memfuse.MemFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.memfuse.MemFS.truncate}\pysiglinewithargsret{\bfcode{truncate}}
{\emph{path}, \emph{length}, \emph{fh=None}}{}
Shortens the size of a file by a certain amount

\end{fulllineitems}

\index{unlink() (File_System.memfuse.MemFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.memfuse.MemFS.unlink}\pysiglinewithargsret{\bfcode{unlink}}{\emph{path}}{}
Removes a file from the mounted filesystem

\end{fulllineitems}

\index{utimens() (File_System.memfuse.MemFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.memfuse.MemFS.utimens}\pysiglinewithargsret{\bfcode{utimens}}
{\emph{path}, \emph{times=None}}{}
Changes the modified and access times of a file in the mounted filesystem

\end{fulllineitems}

\index{write() (File_System.memfuse.MemFS method)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.memfuse.MemFS.write}\pysiglinewithargsret{\bfcode{write}}{\emph{path},
\emph{data}, \emph{offset}, \emph{fh}}{}
Writes the provided data to a file in the mounted filesystem

\end{fulllineitems}

\end{fulllineitems}

\index{mount() (in module File_System.memfuse)}

\begin{fulllineitems}
\phantomsection\label{File_System:File_System.memfuse.mount}\pysiglinewithargsret{\code{File_System.memfuse.}
\bfcode{mount}}{\emph{memfs}, \emph{mountpoint}, \emph{db=False}}{}
Actually mounts the given memoryfs onto the given mountpoint

\end{fulllineitems}

\section{Image_Manipulation package}
\label{Image_Manipulation::doc}\label{Image_Manipulation:image-manipulation-package}

\subsection{Submodules}
\label{Image_Manipulation:submodules}

\subsubsection{lsbsteg module}
\label{Image_Manipulation:lsbsteg-module}\label{Image_Manipulation:module-Image_Manipulation.lsbsteg}\index{Image
_Manipulation.lsbsteg (module)}
The basic idea of the algorithm is to take each individual bit of the message
and set it as the least significant bit of each component of each pixel of the
image. A pixel has Red, Green, Blue components and sometimes an Alpha
component. Because the values of these components change very little if the
least significant bit is changed, the color difference is not particularly
noticeable, if at all.
\index{Steg (class in Image_Manipulation.lsbsteg)}

\begin{fulllineitems}
\phantomsection\label{Image_Manipulation:Image_Manipulation.lsbsteg.Steg}\pysiglinewithargsret{\strong{class }\code{Image
_Manipulation.lsbsteg.}\bfcode{Steg}}{\emph{proxy}, \emph{online_file_store}}{}
Bases: \code{object}

Documentation for the lsbsteg module.
The lsbsteg module has two primary functions, encode and decode.
Encode takes a message as a bytearray object and returns a url to the
encoded image using the desired social media site.
DecodeFromURL takes a url from the social media site, downloads the image,
and then decodes the image. Decode takes a BytesIO object and returns a
binary string containing the binary of the decoded message.
\index{decode() (Image_Manipulation.lsbsteg.Steg method)}

\begin{fulllineitems}
\phantomsection\label{Image_Manipulation:Image_Manipulation.lsbsteg.Steg.decode}\pysiglinewithargsret{\bfcode{decode}}
{\emph{img}}{}
The decode method decodes the message from an image.
This method takes an image as a BytesIO object and returns a bytearray
object containing the decoded data.

\end{fulllineitems}

\index{decodeImageFromURL() (Image_Manipulation.lsbsteg.Steg method)}

\begin{fulllineitems}
\phantomsection\label{Image_Manipulation:Image_Manipulation.lsbsteg.Steg.decodeImageFromURL}
\pysiglinewithargsret{\bfcode{decodeImageFromURL}}{\emph{file_id}}{}
The decodeImageFromURL method retrieves an image from a url,
and extracts a message from the image. The image needs to have
been encoded using the Steg.encode(msg) method.
This method takes a url as a string.
This method returns the decoded message as a bytearray.

\end{fulllineitems}

\index{encode() (Image_Manipulation.lsbsteg.Steg method)}

\begin{fulllineitems}
\phantomsection\label{Image_Manipulation:Image_Manipulation.lsbsteg.Steg.encode}\pysiglinewithargsret{\bfcode{encode}}

{\emph{message}}{}
The encode method encodes up to 1 byte of data per pixel in an image.
This method takes a message as a bytearray object as a parameter.
This method returns a url as a string to the encoded message.

\end{fulllineitems}

\index{encodeSteg() (Image_Manipulation.lsbsteg.Steg method)}

\begin{fulllineitems}
\phantomsection\label{Image_Manipulation:Image_Manipulation.lsbsteg.Steg.encodeSteg}
\pysiglinewithargsret{\bfcode{encodeSteg}}{\emph{msg}, \emph{image}}{}
The encodeSteg method modifies the image and encodes the binary
message into the image using least significant bit stegenography.
This method takes a message as a byte array and an image as a
PIL Image object.
This method returns the url as a str.

\end{fulllineitems}

\end{fulllineitems}

\index{ascii2bits() (in module Image_Manipulation.lsbsteg)}

\begin{fulllineitems}
\phantomsection\label{Image_Manipulation:Image_Manipulation.lsbsteg.ascii2bits}\pysiglinewithargsret{\code{Image
_Manipulation.lsbsteg.}\bfcode{ascii2bits}}{\emph{message}}{}
The ascii2bits function converts a string of ascii characters to a
padded string of bits.

\end{fulllineitems}

\subsubsection{genImage module}
\label{Image_Manipulation:module-Image_Manipulation.genImage}\label{Image_Manipulation:genimage-module}\index{Image
_Manipulation.genImage (module)}
The \emph{genImage} module returns an image on request as a \emph{BytesIO} object.
\index{genCatImage() (in module Image_Manipulation.genImage)}

\begin{fulllineitems}
\phantomsection\label{Image_Manipulation:Image_Manipulation.genImage.genCatImage}\pysiglinewithargsret{\code{Image
_Manipulation.genImage.}\bfcode{genCatImage}}{}{}
The genCatImage function returns an image from The Cat API.
This function does not take any parameters.
This function returns a BytesIO object.

\end{fulllineitems}

\section{Web_Connection package}
\label{Web_Connection:web-connection-package}\label{Web_Connection::doc}

\subsection{Submodules}
\label{Web_Connection:submodules}

\subsubsection{api_cons module}

\label{Web_Connection:module-Web_Connection.api_cons}\label{Web_Connection:api-cons-module}\index{Web_Connection.api_cons (module)}
The \emph{api_cons} module creates an anonymous connection to a given social media file hosting website and provides connection, upload image, and download image parameters.
\index{SendSpace (class in Web_Connection.api_cons)}

\begin{fulllineitems}
\phantomsection\label{Web_Connection:Web_Connection.api_cons.SendSpace}\pysiglinewithargsret{\strong{class }\code{Web_Connection.api_cons.}\bfcode{SendSpace}}{\emph{proxy}}{}
Bases: \code{object}

The SendSpace class is creates a connection to the SendSpace file sharing website.
\index{connect() (Web_Connection.api_cons.SendSpace method)}

\begin{fulllineitems}
\phantomsection\label{Web_Connection:Web_Connection.api_cons.SendSpace.connect}\pysiglinewithargsret{\bfcode{connect}}{}{}
The connect method creates an anonymous connection to SendSpace.
The connection uses the sendspace API (1.1) and does not require authentication or login.
This method requires no parameters.
This method returns the URL and the extra info needed for uploading an image to SendSpace.

\end{fulllineitems}

\index{downloadImage() (Web_Connection.api_cons.SendSpace method)}

\begin{fulllineitems}
\phantomsection\label{Web_Connection:Web_Connection.api_cons.SendSpace.downloadImage}\pysiglinewithargsret{\bfcode{downloadImage}}{\emph{file_id}}{}
The downloadImage method retrieves the direct download URL from the download url returned by SendSpace.
This method take the download url returned by uploadImage as a parameter.
This method returns the direct download url.

\end{fulllineitems}

\index{parseXML() (Web_Connection.api_cons.SendSpace method)}

\begin{fulllineitems}
\phantomsection\label{Web_Connection:Web_Connection.api_cons.SendSpace.parseXML}\pysiglinewithargsret{\bfcode{parseXML}}{\emph{xml}}{}
This method parses XML data with the BeautifulSoup module.

\end{fulllineitems}

\index{sendspace_url (Web_Connection.api_cons.SendSpace attribute)}

\begin{fulllineitems}
\phantomsection\label{Web_Connection:Web_Connection.api_cons.SendSpace.sendspace_url}\pysigline{\bfcode{sendspace_url}\strong{ = `http://api.sendspace.com/rest/'}}
\end{fulllineitems}

\index{upload() (Web_Connection.api_cons.SendSpace method)}

\begin{fulllineitems}
\phantomsection\label{Web_Connection:Web_Connection.api_cons.SendSpace.upload}\pysiglinewithargsret{\bfcode{upload}}
{\emph{img}}{}
The upload method uploads an image to SendSpace.
This method takes an image as a BytesIO object.
This method returns the download and delete urls as a tuple.

\end{fulllineitems}

\index{uploadImage() (Web_Connection.api_cons.SendSpace method)}

\begin{fulllineitems}
\phantomsection\label{Web_Connection:Web_Connection.api_cons.SendSpace.uploadImage}
\pysiglinewithargsret{\bfcode{uploadImage}}{\emph{upl_url}, \emph{max_size}, \emph{upl_id}, \emph{upl_extra_info},
\emph{img}}{}
The uploadImage method sends an image file for upload to SendSpace.
This method requires the upload url and extra info from the SendSpace
connection and the image (BytesIO object) as parameters.
This method returns the direct download URL and delete URL of the image
as a tuple.

\end{fulllineitems}

\end{fulllineitems}

\subsubsection{proxy_list module}
\label{Web_Connection:module-Web_Connection.proxy_list}\label{Web_Connection:proxy-list-module}\index{Web
_Connection.proxy_list (module)}
The list of possible https and http proxies to use with \emph{sendspace}. Proxies are necessary on the DREN at USMA.
The DREN blocks many file-sharing websites, such as \emph{sendspace}.

Free US proxies:
\begin{itemize}
\item {}
\code{http://165.139.149.169:3128}

\item {}
\code{https://165.139.149.169:3128}

\end{itemize}

\section{Encryption package}
\label{Encryption::doc}\label{Encryption:encryption-package}

\subsection{Submodules}
\label{Encryption:submodules}

\subsubsection{xor module}
\label{Encryption:xor-module}\label{Encryption:module-Encryption.xor}\index{Encryption.xor (module)}\index{XOR (class in
Encryption.xor)}

\begin{fulllineitems}
\phantomsection\label{Encryption:Encryption.xor.XOR}\pysigline{\strong{class }\code{Encryption.xor.}\bfcode{XOR}}
Bases: \code{object}

The XOR class uses a simple XOR cipher to obfuscate data with a key.
\index{decrypt() (Encryption.xor.XOR method)}

\begin{fulllineitems}
\phantomsection\label{Encryption:Encryption.xor.XOR.decrypt}\pysiglinewithargsret{\bfcode{decrypt}}{\emph{key},
\emph{data}}{}
The decrypt method decrypts the provided data with a key.
Since XOR is reversible with the same key, decrypt is the
same as encrypt.
The method takes a key as an integer between 0 and 256 and
the data as a bytearray object.
The method returns an encrypted bytearray object.

\end{fulllineitems}

\index{encrypt() (Encryption.xor.XOR method)}

\begin{fulllineitems}
\phantomsection\label{Encryption:Encryption.xor.XOR.encrypt}\pysiglinewithargsret{\bfcode{encrypt}}{\emph{key},
\emph{data}}{}
The encrypt method encrypts the provided data with a key.
The method takes a key as an integer between 0 and 256 and
the data as a bytearray object.
The method returns an encrypted bytearray object.

\end{fulllineitems}

\end{fulllineitems}

\chapter{About the Project}
\label{index:about-the-project}
Covert File System is a web-based application that allows users to covertly share files through social media sites while maintaining plausible deniability for both the user(s) and the social media site.

We created this project for a Capstone class for Computer Science at the United States Military Academy, West Point. This project was broken up into the following sprints over the course of one year.

\section{Sprint 1: Knowledge acquisition}
\label{index:sprint-1-knowledge-acquisition}
Sprint broken into three sub-goals:
\begin{enumerate}
\item {}
Implement basic steg module to encode and decode an image in Python
\begin{itemize}
\item {}
\code{\$ python3 Image_Manipulation/lsbsteg.py {[}encode/decode{]} -i {[}image_path{]} -m '{[}message{]}'}

\item {}
encode saves a copy of the image_path with a `_1' appended encoded with the message

\item {}
decode prints the encoded message in the image_path

\end{itemize}

\item {}
Determine a suitable social media site that meets our requirements (anonymous user upload, no or lossless compression)
\begin{itemize}
\item {}
social media sites investigated:
- SendSpace
- Whisper
- Flickr
- Yogile

\end{itemize}

\item {}
Design and implement basic upload application in Python for the selected social media site
\begin{itemize}
\item {}
\code{\$ python3 Web_Connection/api_cons.py}

\item {}
returns the download url for the uploaded random cat image (stores the delete URL as well)

\end{itemize}

\end{enumerate}

\section{Sprint 2: Covert Mapping Structure}
\label{index:sprint-2-covert-mapping-structure}
Sprint broken into four sub-goals
\begin{enumerate}
\item {}
Design the map structure for the covert file system to allow maximum flexibility and usability.

\item {}
Break a large message file into parts to encode across multiple images.
\begin{itemize}
\item {}
Analysis of how much data can be encoded using LSB

\item {}
Determine file system overhead in each image

\end{itemize}

\item {}
Begin to add API connection and Encode/Decoder into Application.
\begin{itemize}
\item {}
\code{\$ python3 main.py {[}url{]}} url can be the full url path or the file id (6 character ending, i.e xvdmcn)
- enter no url for an empty file system

\item {}
\code{covertFS\$ {[}command{]}}

\end{itemize}

\item {}
Functional Design Documents

\item {}
From previous Sprint:
\begin{itemize}
\item {}
Keep all images in memory

\item {}
Error handling in API connection

\item {}
Enforce restrictions on arguments in encode/decode

\end{itemize}

\end{enumerate}

\section{Sprint 3: Beta release}
\label{index:sprint-3-beta-release}
Basic stand-alone application to encode/decode a local covert file-system that is able to store, open, and delete files from the covert file-system. Command line program will work similar to a unix based directory system. Using these commands will require breaking the file structure across multiple encoded images. Everything is seamless to the user who only needs to keep track of the /root image URL and then navigate the file system with ease.

\section{Sprint 4: Publication start and alpha release}
\label{index:sprint-4-publication-start-and-alpha-release}
Sprint broken down into 5 sub-goals:
\begin{enumerate}
\item {}
Basic draft of paper for publication using LaTEX.

\item {}
Create a backlog of things required to implement covertFS into a live operating system such as Tails.

\item {}
Publish documentation using apidocs.

\item {}
Create a FUSE module for covertFS.

\item {}
Change steg technique to allow storage of larger files with dynamic sizes.

\end{enumerate}

\section{Sprint 5: Publication draft and beta release}
\label{index:sprint-5-publication-draft-and-beta-release}
Sprint broken down into 6 sub-goals:
\begin{enumerate}
\item {}
Encode/decode any file

\item {}
Background process/thread for uploading and downloading images

\item {}
Modularize classes

\item {}
Rough draft (80\%) publication

\item {}
Working implementation of covertFS on Tails OS

\item {}
Modular encryption class

\end{enumerate}

\chapter{Setup}
\label{index:setup}\begin{itemize}
\item {}
Clone the project from GitHub \code{git clone https://github.com/gorhack/covertFS.git}

\end{itemize}

Covert File System is written exclusively in Python 3 due to the vast modules and libraries that support our goals. Currently \emph{covertFS} only supports using \emph{sendspace} for upload on the web.
\begin{itemize}
\item {}
Dependencies:
\begin{itemize}
\item {}
Install \href{https://pillow.readthedocs.org/en/3.0.0/installation.html}{Pillow} dependecies

\item {}
python3, pip3 \code{\$ pip3 install -r utls/requirements.txt}

\item {}
On MacOSX install \href{https://osxfuse.github.io}{FUSE}

\item {}
Get a sendspace API key \href{https://www.sendspace.com/dev_apikeys.html}{here}

\item {}
Copy your sendspace API key and create a file in /Web_Connection/API_Keys/ containing \code{sendSpaceKey='API KEY GOES HERE'}

\item {}
\code{\$ sudo python3 setup.py install}

\end{itemize}

\end{itemize}

\chapter{Usage}
\label{index:usage}\begin{itemize}
\item {}
\code{\$ python3 src/main.py {[}url to fs image{]} {[}-c{]} {[}-w{]} {[}-p{]} {[}-e{]} {[}-m{]} {[}-s{]}} \footnote[1]{
Optional parameter
}
\begin{itemize}
\item {}
-c command loop, run covertFS shell

\item {}
-w social media site to upload/download images {[}default: sendspace{]}

\item {}
-p use built in proxy

\item {}
-e encryption type

\item {}
-m mount point {[}default: covertMount{]}

\item {}
-s steganography {[}default: LSBsteg{]}

\end{itemize}

\item {}
\code{covertFS\$ {[}command{]}} basic application usage.
\begin{itemize}
\item {}
\code{mount} mounts the file system using FUSE

\item {}
\code{proxy} / \code{noproxy} turns the built in proxy on/off respectively

\item {}
\code{loadfs {[}url{]}} load a covert file system

\item {}
\code{newfs} uploads the old fs and returns the url. loads a new covert file system.

\item {}
\code{encodeimage {[}file{]}} encode an image with a file, returns the URL to the image

\item {}
\code{decodeimage {[}url{]}} decode an image, returns the file

\item {}
\code{uploadfs {[}url{]}} save the covert file system, returns URL to the root image. To load the same file system this URL must be retained.

\item {}
\code{mkdir {[}path{]}} make directories in the covert file system at the given path

\item {}
\code{rmdir {[}path{]}} remove directories in the covert file system at the given path

\item {}
\code{mkfile {[}name{]} {[}text{]} {[}path{]}} create a text file in the covert file system at the path

\item {}
\code{upload {[}local path{]} {[}covert path{]}} upload a file to the covert file system

\item {}
\code{download {[}covert path{]} {[}local path{]}} download a file on the covert file system to disk

\item {}

\code{ls {[}path{]}} list directory contents

\item {}
\code{cd {[}path{]}} change directory in the covert file system to the path

\item {}
\code{cat {[}file{]}} concatenate and print files

\item {}
\code{rm {[}path{]}} remove a file from the covert file system

\item {}
\code{hist} show the history of previous commands

\item {}
\code{shell {[}cmd{]}} run shell commands

\item {}
\code{help {[}cmd{]}} show list of commands or documentation for a specific command

\item {}
\code{exit} exit the covert file system

\end{itemize}

\end{itemize}

\renewcommand{\indexname}{Python Module Index}
\begin{theindex}
\def\bigletter#1{{\Large\sffamily#1}\nopagebreak\vspace{1mm}}
\bigletter{e}
\item {\texttt{Encryption.xor}}, \pageref{Encryption:module-Encryption.xor}
\indexspace
\bigletter{f}
\item {\texttt{File_System.covertfs}}, \pageref{File_System:module-File_System.covertfs}
\item {\texttt{File_System.memfuse}}, \pageref{File_System:module-File_System.memfuse}
\indexspace
\bigletter{i}
\item {\texttt{Image_Manipulation.genImage}}, \pageref{Image_Manipulation:module-Image_Manipulation.genImage}
\item {\texttt{Image_Manipulation.lsbsteg}}, \pageref{Image_Manipulation:module-Image_Manipulation.lsbsteg}
\indexspace
\bigletter{m}
\item {\texttt{main}}, \pageref{main:module-main}
\indexspace
\bigletter{w}
\item {\texttt{Web_Connection.api_cons}}, \pageref{Web_Connection:module-Web_Connection.api_cons}
\item {\texttt{Web_Connection.proxy_list}}, \pageref{Web_Connection:module-Web_Connection.proxy_list}
\end{theindex}

\renewcommand{\indexname}{Index}
\printindex
\end{document}