PyCIRCLean: a versatile Python framework to check and/or sanitize files

CIRCL
Computer Incident
Response Center
Luxembourg

TLP:WHITE

info@circl.lu

March 10, 2017

Overview

- Aims to be used in dedicated security applications to sanitize documents from hostile to trusted environments.
- Generic way to handle large collections of files
- Generate audit logs
- Comes with many helpers
- Defensive programming

Implementation

- Copies files from a directory (source) to an other one (destination)
- Computes hashes (sha256) of all the files in the source
- Creates a directory tree on the destination directory
- Gets the mime type of each file

Logging and reporting

- Every processing is logged
- Medatata (filetype, size, name, extension, ...) are kept
- Any error occurring during the processing is stored
- WiP: generating a human readable report (Markdown, HTML)

Action of the main script

- Discard known extensions with active content
- Verifies if the extension corresponds to the mimetype (polyglot files)
- Force extension on supposedly text files
- Discards windows executables
- Discard Office (Libreoffice and Windows Office) document with active content
- Discard PDFs with active content
- Unpack archives and process content
- Extract metadata from images

Plus / Minus

- Plus
  - (almost) Pure python
  - Reliable
  - Fast
- Minus
  - Does not block a 0 day in a non-active content
  - Medium level of false positive (non-malicious active content)

Implement your own module - FileBase

- The default constructors gets the mime type of the file and initialize the log of the file
- Surcharge the constructor accordingly to your needs
- Has helpers to get and set information on the file being processed
- Can force the extension of the file when copied
- All those functions have to be used in order to handle the files accordingly to your requirements

Implement your own module - KittenGroomerBase

- The default constructor cleans the destination directory
- Starts the general logging
- Iterate through all the files on the src key
- Has helpers to handle safely the file management

Implement your own module - GroomerLogger

- The default constructor initialize the log files
- Creates a tree representation of the content, computes the hashes
- Stores the logs for each processed file

Hardware implementation - RaspberryPi

- Standalone device
- Easy to carry around
- Not used for anything else
- Cheap and easy to setup

Security considerations

- Assuming the content might be malicious
- Parsing is very vulnerable to exploits
- Unpacking archives and recursion need to stop (halting problem)
- KISS, default features and ease to update
- Distrust everything (your code, and other people's code)

Defensive programing - Questions

- How can an attacker interact with the code? With the device?
- What are the most critical part of the project?
- How to handle unexpected behavior?
- What happen if there is an unpatched vulnerability?

Defensive programming - Remediations

- Bare Debian for Raspberry
- Few dependencies
- Image read only
- Code runs as user
- Small code base