Implementation

How do we demonstrate encryption? How can we test encryption?

⚠ Download Research

Project Roadmap



IDEA

Compare encryption algorithms.



ENCRYPTION

Identify encryption algorithms.



OBJECTIVES

Create encyption tests.



Performs tests for data.



DISCOVERIES

Present findings & comparisons.



CONCLUSION

Determine the best encryption.

Steps to Implement

AES and RSA.

Create Data

• Using python, I will create a program that takes in a small text file to create 3 separate files of varying size. I also created a python program to count the data and confirm the expected file's size for accuracy.

Create Algorithms

Using Cryptography and Cryptodome modules in python I will create a program to use AES and a program to
use RSA and encrypt each file. The programs will record the start and end time to determin the time-lapse for
analysis.

Create Results

• Within the algorithms, I record the times in a result.txt file labeled for AES and RSA, respectively. The results files are then used to create the charts for measuring speed of encryption.

Types of Comparison

Encryption Speed and Encryption Strength.

- Encryption Speed
 - I will perform 50 tests for each encryption algorithm to determine speed efficiency. Each test takes in three files of differing sizes and encrypts the files while recording the start and end time.
- For these same tests and results I will determin strength of the encrypted algorithm. Without a better metric, I chose to compare the size of the encrypted files created to gauge strength.

Metrics

Number of Tests Speed Simulation Pass/Fail Tests Accuracy Compare Results Examine Hypothesis **Record Conclusions** Publish

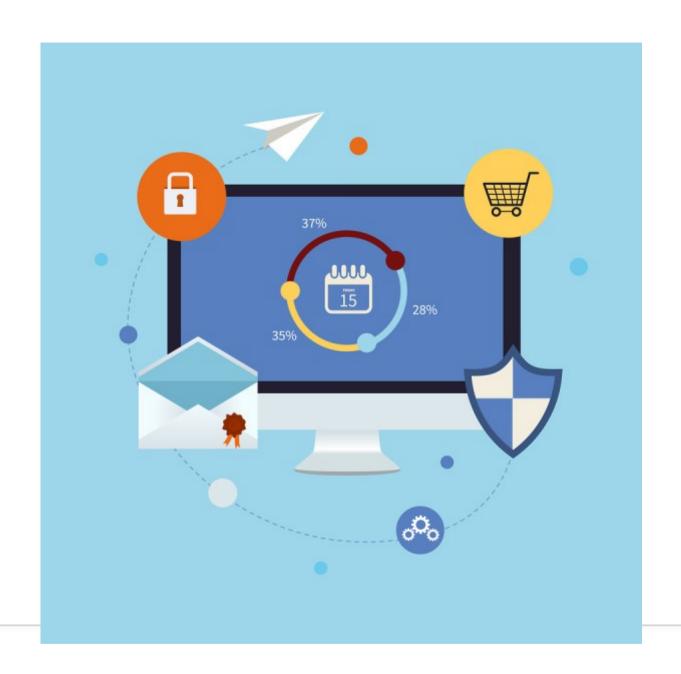
Files on Github.



```
r00t@r00t-KitPloit:~/KitPloit$ uncompyle6 -h
 uncompyle6 [OPTIONS]... [ FILE | DIR]...
uncompyle6 [--help | -h | --V | --version]
  uncompyle6
                       foo.pyc bar.pyc
                                                   # decompile foo.pyc, bar.pyc to stdout
  uncompyle6 -o . foo.pyc bar.pyc # decompile to ./foo.pyc_dis and ./bar.pyc_dis uncompyle6 -o /tmp /usr/lib/python1.5 # decompile whole library
Options:
                    output decompiled files to this path: if multiple input files are decompiled, the common prefix
  -o <path>
                    is stripped from these names and the remainder appended to
                      uncompyle6 -o /tmp bla/fasel.pyc bla/foo.pyc

-> /tmp/fasel.pyc_dis, /tmp/foo.pyc_dis

uncompyle6 -o /tmp bla/fasel.pyc bar/foo.pyc
                      -> /tmp/bla/fasel.pyc_dis, /tmp/bar/foo.pyc_dis
uncompyle6 -o /tmp /usr/lib/python1.5
-> /tmp/smtplib.pyc_dis ... /tmp/lib-tk/FixTk.pyc_dis
   --compile | -c <python-file>
                    attempts a decompilation after compiling <python-file>
                    print timestamps
  -p <integer> use <integer> number of processes
                    recurse directories looking for .pyc and .pyo files
                   use fragments deparser
  --fragments
                    compare generated source with input byte-code
  --verify-run compile generated source, run it and check exit code
  --syntax-verify compile generated source
--linemaps generated line number correspondencies between byte-code
                    and generated source output
  --encoding <encoding> use <encoding> in generated source according to pep-0263
  --help
                    show this message
 ebugging Options:
  --asm
                 -a include byte-code
                                                      (disables --verify)
  --grammar
                      show matching grammar
  --tree
                      include syntax tree (disables --verify add template rules to --tree when possible
                                                     (disables --verify)
  --tree++
Extensions of generated files:
   .pyc_dis' '.pyo_dis'
+ '_unverified'
                                successfully decompiled (and verified if --verify) successfully decompile but --verify failed
     + ' failed'
                                 decompile failed (contact author for enhancement)
 r00t@r00t-KitPloit:~/KitPloit$ ∏
    KitPloit : bash
IMPLEMENTATION
```



CONCLUSION

Closing Remarks. 🖫

There are only two kinds of companies:

- 1. Those that have been hacked.
- 2. Those that will be.
- ~ Robert Mueller

Best,

Brandon Rowe