MysteryTwister C3

THE CRYPTO CHALLENGE CONTEST

Kaskade-S/T – Part 6 (RansomKaskade Part 1)

Author: Miroslav Dimitrov / Bernhard Esslinger

June 2017

Introduction (1/2)

Honeypots used by antivirus companies were triggered by a new ransom virus. When doing the reverse engineering it appears that the virus is using a modified version of the Kaskade S/T algorithm.

The cryptography experts in the company decided adding a proactive layer of security by implementing a "trap" for the ransom cryptovirus:

a) During the update of the antivirus software an extra folder is created. This folder contains a special set of files.

b) If the machine is infected in the future, the cryptovirus will encrypt those files too.

c) The files were specially designed to include all the possible paddings.

d) The experts hope to recover the keys with automatic cryptanalysis.

e) Having the keys, they can recover all the files in the victim's machine.

Introduction (2/2)

In this challenge, you get such a set of encrypted "trapped" files in a directory called Chunks with over 300 files.

Each file chunk[0..307].txt had a specific length between 256 and 512 bytes. All the encrypted files, but one, have a length of 512 bytes.

The padding in the RansomKaskade algorithm is done by just filling the plaintext files with "X" until it reaches a length of a multiple of 256 bytes.

Only speculations could be made why the attackers used the RansomKaskade algorithm instead of a modern symmetric cipher. Maybe they want to have an implementation which is platform independent or maybe they didn't trust implementations where a secret service may have trapped the implementation being delivered with the operating system.

Challenge (1/2)

In the new variant of the Kaskade S/T algorithm, the permutations have – unlike as in Part 1 to 5 – different lengths, and the transposition works on bits instead of bytes:

- The subst key has 256 elements from 0 to 255.
- The transp key has 2048 elements from 0 to 2047.

The additional zip file of this challenge contains all the encrypted files in the Chunks directory.

Furthermore there is a Python 3 program, with which you can test the decryption and encryption (see page 6).

Challenge (2/2)

Could you reveal the unknown key – the same one was used to encrypt all files. You have to perform a fully known-plaintext attack as you know both the plaintext and the ciphertext files in the Chunks directory.

The key used consists of two random permutations, one for the substitution and one for the transposition. To solve the challenge you need the complete and correct key.

To get the solution please decrypt the given file "passphrase.enc" with the key you found. Please enter it in lower-case letters and without spaces or punctuation.

RansomKaskade.py

With the given Python program you are able to test whether the given ciphertexts can be decrypted correctly. Here is an example:

The call for encryption is:
python RansomKaskade.py -e -i plain.txt -k randomkey -o encrypted.enc

The call for decryption is:
python RansomKaskade.py -d -i encrypted.enc -k randomkey -o revealed.txt

Additional Files

The additional zip archive contains the following:

- directory "Chunks" (contains all the encrypted files and all the according plaintext files)
- file "passphrase.enc" (encrypted passphrase for the solution)
- file "RansomKaskade.py"
- file "testkey" (a typical key generated by RansomKaskade.py with the option -r -k)