

# Parallel Outlook Algorithm Collider

---

## Abstract

this is a demonstrative software that find collisions in the algorithm that provide protection to the Outlook .pst files password taking advantage of all the resources of parallel hardware (multicore,manycore,multiprocessor).

## The Bug

So, let's begin with the point that Microsoft Office Outlook's .PST file is file-type data storage on a local computer, which stores contacts, notes, e-mail messages, and other items arranged by a certain order. A .PST file can be used as a default location for delivering e-mail messages. It can be also used for ordering and backing up data.

To protect the content of a .PST file and restrict the unauthorized access to it by third parties, one can set a password of up to 15 characters long. In such a case, it is commonly thought that the .PST file cannot be opened unless one knows the original password. Let's see how true that is.

A .PST file access password is neither remembered nor stored in an explicit form. Instead, the computer calculates the password's hash value and stores it in the .PST file or, if the option 'Save this password in your password list' is selected, in Windows Registry, encrypted additionally.

The most interesting thing is that the password hash calculation algorithm is not the actual hashing algorithm - it is rather a simple CRC32 checksum calculation routine. CRC32 is a redundancy check algorithm, and it surely isn't a hashing routine. For one or another reason, Microsoft had decided to use that algorithm instead of stronger ones like SHA. Keeping it as inheritance from older versions of Outlook, MS has not been changing the algorithm for long time, being led, as it seems, by backward compatibility thoughts. On the other hand, it is still unclear, why that wasn't changed when Outlook 2003 was released (Outlook 2003's .PST format has 64-bit internal addressing and is not compatible with the previous versions.

Yet another interesting observation: if a checksum stored in a .PST file is equal to zero, the program 'thinks' that no password is set. However, since we know that passwords with the same checksums do occur, we can suppose there are passwords, which checksums are also equal to zero. Such passwords indeed exist. Here is a short portion of the large list: 1Rj78C, 5J8j84, ArTniW.  $\text{Crc32}("1Rj78C")=0$ ,  $\text{Crc32}("5J8j84")=0$ ,  $\text{Crc32}("ArTniW")=0$ . If we set one of such passwords to protect our .PST file, we will actually have that file unprotected, and whenever someone tries to access it next time, it will not even prompt for a password. Don't believe it? - Try it yourself.

An experiment has shown that on average it takes about a minute to recover an Outlook hash password using the brute force attack. However, the crypto analysis of CRC32 has revealed that the algorithm is completely reversible for short passwords (up to 4 characters) and partially reversible for all others. That means, one can recover the original password or its CRC32 equivalent password, that will be indistinguishable for Outlook, almost instantly. It has been proven that it requires not more than 7 characters to pick a collision (password with the same checksum as the original password).

## The technology

For the development of this software they have been used the Microsoft Parallel Extension for .NET Framework 3.5.

Parallel Extensions to the .NET Framework is a managed programming model for data parallelism, task parallelism, and coordination on parallel hardware unified by a common work scheduler.

Parallel Extensions was designed to provide performance gains on multiple core or multiple processor machines. Instead of making decisions during development that limit the upper bound of potential parallelism, much of the library scales automatically to utilize more cores as they become available.