

ChaOS - TNT 1.0

Official Documentation

Contents

[0 Overview 3](#_Toc130908388)

[0.0 Commands 4](#_Toc130908389)

[0.0.0 CMD create 4](#_Toc130908390)

[0.0.1 CMD read 4](#_Toc130908391)

[0.0.2 CMD delete 4](#_Toc130908392)

[0.0.3 CMD burn 5](#_Toc130908393)

[0.0.4 CMD restore 5](#_Toc130908394)

[0.0.5 CMD edit 5](#_Toc130908395)

[0.0.6 CMD dir 5](#_Toc130908396)

[0.0.7 CMD echo 6](#_Toc130908397)

[0.0.8 CMD clear 6](#_Toc130908398)

[0.0.9 CMD help 6](#_Toc130908399)

[0.0.10 CMD shutdown 6](#_Toc130908400)

[0.0.11 CMD whoami 6](#_Toc130908401)

[0.0.12 CMD syslog 6](#_Toc130908402)

[0.0.13 ipconfig 7](#_Toc130908403)

[0.1 TNTFS 8](#_Toc130908404)

[0.1.0 Technical overview 8](#_Toc130908405)

[0.1.0.0 Access permissions 8](#_Toc130908406)

[0.1.0.1 Directory types 8](#_Toc130908407)

[0.1.0.2 Account types 9](#_Toc130908408)

# 0 Overview

As the name implies, ChaOS’s source code is incredibly chaotic and inconsistent. The main procedure of the program goes as follows:

Upon startup, the user is presented a login prompt in the console. After successful login, command\_prompt() is called:

This is the main interface throughout the whole program, comparable to other text-based UIs like DOS.

A:/Users/kaf221122>

From now on, the user is free to execute commands (see chapter [1.1 Commands](#_1.1_Commands) for more information.)

## 0.0 Commands

The following commands are implemented in ChaOS so far:

### 0.0.0 CMD create

Is used to create files, directories and other users. Can be used with the keyword cr.

**Syntax:**

File:

cr f <filename>

Dir:

cr d <dirname> (<access\_permission>)

User:

cr u

### 0.0.1 CMD read

Is used for read-only display of txts. Also shows meta information like creator, time of creation and ChaOS version. Can be used with the keyword rd.

**Syntax:**

rd f <filename>

### 0.0.2 CMD delete

Is used for moving files and directories into the recycling bin (given the source location was in a non-system directory like A: or A:/Users). Users can be deleted too, but will face immediate termination, not temporary storage in the recycling bin. Can be used with the keyword del.

**Syntax:**

File:

del f <filename>

Dir:

del d <dirname>

User:

del u <username>

### 0.0.3 CMD burn

Is used to delete a file or directory without temporary storage in the recycling bin. Note that burning the recycling bin will permanently delete every file or directory inside but won’t remove the recycling bin directory itself.

**Syntax:**

Fil:

burn f <filename>

Dir:

burn d <dirname>

### 0.0.4 CMD restore

Is used for restoring files or directories from the recycling bin in the current directory. Only works in non-system directories. Can be used with the keyword res.

**Syntax:**

File:

res f <filename>

Dir:

res d <dirname>

### 0.0.5 CMD edit

Is used to write to a file (currently only txts) or to change user details like name, password, or account type. Can be used with the keywords alt or ed.

**Syntax:**

File:

alt f <filename>

User:

alt u <username>

### 0.0.6 CMD dir

Is used to display all files or and directories in the current directory. Can be used with the keyword d.

**Syntax:**

d

### 0.0.7 CMD echo

Is used to echo a string, there’s no apparent reason for this command’s existence.

**Syntax:**

echo <str>

### 0.0.8 CMD clear

Is used to clear the console of any past in- and outputs. Can be used with the keyword cl

**Syntax:**

cl

### 0.0.9 CMD help

Is used to get short summarisation and syntax for a command if you’re too lazy to read the documentation.

**Syntax:**

help <cmd>

### 0.0.10 CMD shutdown

Is used to shutdown the system. Can be used with the keyword sd.

**Syntax:**

sd

### 0.0.11 CMD whoami

Is used to display the username and hostname. More information coming soon.

**Syntax:**

whoami

### 0.0.12 CMD syslog

Is used to display all performed actions (executed commands, creations, deletions, alterations etc.) in the current session. Resets after shutdown.

**Syntax:**

syslog

### 0.0.13 CMD ipconfig

Is used to display information about the current network and client.

**Syntax:**

Compact:

ipconfig

Detailed:

ipconfig all

### 0.0.14 CMD move

Is used to move files or directories to a new location. Can be used with directory- and filenames (for moving files inside the current directory) or with fully qualified paths (for moving objects in different parent directories). Can be used with the keyword mv.

**Syntax:**

File:

mv f <source> <destination>

directory:

mv d <source> <destination>

## 0.1 TNTFS

The **T**otally **N**ew **T**echnology **F**ile **S**ystem is the core of ChaOS’s file handling. It allows for specific access permissions for directories which can be specified by the creator. The newest current version, TNTFS 2.0 brings the most crucial features. It is planned to add more specific permissions for both directories and files in the future.

### 0.1.0 Technical overview

As already mentioned above, TNTFS allows for access permissions. It also records other metadata for each directory in the A: drive. This is accomplished with the metadata.csv files in every exiting parent directory. The required information (name, owner, owner account type, access permissions, directory type) for each directory are saved in said metadata.csv file in the parent directory.

For example: the metadata for A:/Users/kaf221122/Desktop/personal\_dir is saved in A:/Users/kaf221122/Desktop/metadata.csv.

Here’s what one of these files looks like:

dirname,owner,owner\_account\_type,access\_permission,dir\_type  
  
documents,kaf221122,dev,kaf221122,personal  
  
desktop,kaf221122,dev,kaf221122,personal  
  
recycling bin,kaf221122,dev,kaf221122,personal

### 0.1.0.0 Access permissions

There are three different types of access permissions in TNTFS: permission for a single specific user, permission for a specific account type and permission for all users.

### 0.1.0.1 Directory types

TNTFS distinguishes between personal, communist, and capitalist directories.  
Personal directories are the autogenerated standard directories, communist directories allow access for all users and capitalist directories for a single user. It may seem confusing to have implemented this alongside the access permissions and yes it is, it adds to the chaos factor, which is incredibly important to keep ChaOS chaotic.

### 0.1.0.2 Account types

There are currently three different account types in ChaOS:

Standard

Has the basic abilities to interact with the system. Has permission to create, delete and alter files and directories inside his directories.

Admin

Has more advanced permissions, including creating and deleting other users or their files and directories.

Developer

The highest in the account type hierarchy. Developers have unlimited access to the system. They capable of resetting the entire A: drive or user data, and deleting system files. The developer also has access to the ChaOS DevTools.

# 1 Technical Overview

ChaOS is 100% written in Python 3. It is largely coded functionally, which the development team is planning to change in the near future. As the name implies, it is a giant steamy pile of inconsistent spaghetti code.

## Main procedure

ChaOS is a console-based OS, which obviously only allows for linear interaction with the system. Here’s what happens from command to execution:

### 1.0.0 Command interpretation

Upon startup of the system, the user is prompted for his username or account creation. After logging in, he has access to the full functionality of ChaOS (See [0.0 Commands](#_0.0_Commands) for more information).

A typical command in ChaOS mainly consists of three to four separate parts:

Ein Bild, das Diagramm, Text enthält.

Automatisch generierte Beschreibung

In this case, the command would be create, the target object a directory, the secondary statement the directory name and the ternary statement the access permissions.

Inside of command\_prompt() the command is first converted into a list, then validated for detectable errors. After that, the command is checked for a sudo statement. If so, it’s removed from its current index to the last index in the list cmd\_split. Then, the command is translated from UI language by the user to system language with translate\_command(). For example, the command create is translated to the keyword cr or a filename with no filetype extension is added a .txt extension. Finally, cmd\_map and args\_map are checked for an occurrence of the entered command, if so, execute\_cmd() is called, else, the remaining commands which aren’t present in the cmd\_map(), are tried and if no match is detected, an error message is displayed.