1. Introduction
This document is the official standard of PWDDB (Password Database) extension. This extension is used to store a big amount of passwords in a single database. This extension has been designed specifically for this, so it is very efficient. It is very fast to read it. In passwords, you can store every ascii character, so that means 256 possibilities per character. You can store passwords of length up to 255 bits (excluding compression).

2. Licence

3. Definition of terms

3.1 Lower bound character
The lower bound character is the ascii code of the minimal character allowed in passwords. It is used for compression.

3.2. Higher bound character
The higher bound character is the ascii code of the maximal character allowed in passwords. It is used for compression.

3.3 Name depth
The maximum number of bytes of each names.

3.4 Password depth
The number of bits of each password.

4. Structure

4.1. Introduction
This part describes how the entension is structured.

4.2 PWDDB signature
The first 6 bytes of all PWDDB files should be :

Binary :

1010000	1010111	1000100	1000100	1000010	1000000

Decimal :

80	87	68	68	66	64

Hexadecimal :

50	57	44	44	42	40

4.3 Meta Data Chunk
This part is delimited by symbols SHEAD (for the start) and EHEAD (for the end). The first byte is lower bound character, the second one is the higher bound character. Then, there is one byte for the name depth and one for the password depth. They should both be different to 0.

4.4 Names Chunk
This part is delimited by symbols SNAME (for the start) and ENAME (for the end). It contains every name, chained without any separator. The number of characters of a name should not exceed the name depth. If the length is not great enough, you should add as many 0 as necessary at the end of the name.

4.5 Passwords Chunk

This part is delimited by symbols SPWD (for the start) and EPWD (for the end). It contains every password, chained without any separator. The size of every password should be exactly equal to the password depth. A byte can contain two parts of 2 different passwords if the password depth is not a multiple of 8. Characters are stored with the formula : char_code – lower_bound_character. The number of bits used for each character is the lower power of 2 that is greater or equal to higher_bound_character – lower_bound_character.

4.6 Connect names and passwords

The way to connect names and passwords cannot be easier : the first name is the name of the first password, the second name is the name of the second password, … So there should be as many names as passwords.