bashacks Documentation

Release 2.0.0

Fernando Merces, Wesley Leite

May 10, 2022

Contents

Bashacks is an open source (GPL) set of bash functions probably useful for programmers, security analysts and general users that need to do some low level type of operation.

In fact, there nothing really new bashacks since all functions are write using exiting software in GNU/Linux distribuitions.

But you still can have advantage in use short commands to run tasks that commomnly will require a ot of lines to be done.

Contents:

CHAPTER 1

Install

In this section you can find instructions bashacks's installation process, if you have any tips to improve, send your opinion. ;)

1.1 Requirements

- bash >= 4
- bc
- binutils
- coreutils
- curl
- file
- grep
- hexdump
- html2text
- perl
- sed
- wget
- xxd
- zip
- make

We consider that your system has the minimum instalation of packages, if you are using some of the below no problems.

```
GNU/Linux
FreeBSD (new)
OSX (new)
```

1.2 Download

GIT

We recommend the version available on `MASTER REPOSITORY ON THE GITHUB`, however, if you want to use the development version skip the session that describe the steps of the GIT

Download the final version, download on the `GITHUB`. Download Bashacks

1.3 GIT

Since the end of version `1.5.0` we are working with separate branches for various activities, can have errors and problems in `devel` repository, but try to keep always clean `master` branch to your advantage.

```
$ git clone https://github.com/merces/bashacks.git
$ cd bashacks
# this's a optional mode, maybe there's some error in the devel branch.
$ git checkout devel
```

1.4 Compile

```
Options

all       : Just creates the file bashacks.sh
install   : Creates the file bashacks.sh, add entry in /etc/bash.bashrc and install␣
 ↪man page
clean     : Just remove the file bashacks.sh
uninstall : remove reference the source of /etc/bash.bashrc and man page
```

```
$ make all
$ source bashacks.sh
```

Done that all functions can be used use `bh TAB TAB` and has a `Sight beyond sight`

CHAPTER 2

Crypto

In this page, you will found information from all functions of cryptograpy on the bashacks

2.1 bh_hashcrack

This is our old function `bh_unmd5` with many improvements, now can with several hashes, just update the name.

In this new version we cache the hash that have already been found, improving delivery speed.

Note: Usage

`bh_hascrack` [hash string]

Supported hash string for decryption:

`md5`, `sha1`, `sha256`, `sha384`, `sha512`

```
#md5
$ bh_hashcrack e10adc3949ba59abbe56e057f20f883e
123456

#sha1
$ bh_hashcrack 38464bf083d958b53580c63c01e56707fd043588
rock

#sha256
$ bh_hashcrack 9ca0f72f324a7bd2c2efc64b40d1e769a473451c2b9e5dfbd54a9db53c986ba5
mamonas

#sha384
$ bh_hashcrack␣
↪504f008c8fcf8b2ed5dfcde752fc5464ab8ba064215d9c514785180d2ad7cee1ab792ad44798c
1234
```

(continues on next page)

(continued from previous page)

```
#sha512
$ bh_hashcrack␣
↪5b01e57fd8ab53cc7c0d2a97585ba5a9d70f0dc966472b32736c52a4823f3fb43532dfc1e83fd2d92f1a7dbec8c401f4d7
hack
```

This function has given a lot of work, many upgrades, have a good time that we have to find a good source hashed base.

2.2 bh_rot

Encrypts/Decrypts string with the Cesar Cipher using n shifts to the right.

Note: Usage

`bh_rot` [int] [string]

int : Aumount of jumps you want to give to the right

string : string to code or decode

```
$ bh_rot 3 terra
whuud
$ bh_rot 13 terra
green
```

Facilities

We created some facilities, aliases for multiple entries of `rot` function, see below.

```
$ bh_rot13 terra
green
$ bh_rot18 terra
lwjjs
$ bh_rot47 adjust

$ bh_rot5 terra
yjwwf
```

bh_rotall is an implementation that accesses rot generating 1..25 results to rot.

```
$ bh_rotall urfn
ROT1 vsgo
ROT2 wthp
ROT3 xuiq
ROT4 yvjr
ROT5 zwks
ROT6 axlt
ROT7 bymu
ROT8 cznv
ROT9 daow
ROT10 ebpx
```

(continues on next page)

(continued from previous page)

```
ROT11 fcqy
ROT12 gdrz
ROT13 hesa
ROT14 iftb
ROT15 jguc
ROT16 khvd
ROT17 liwce
ROT18 mjxf
ROT19 nkyg
ROT20 olzh
ROT21 pmai
ROT22 qnbj
ROT23 rock
ROT24 spdl
ROT25 tqem
```

2.3 bh_strxor

Calculates exclusive OR of each character in a string with a key.

Note: Usage

`bh_strxor` [key] [string]

key : int or hex

string: string to code or decode

```
$ bh_strxor 15 'hack'
gnld

$ bh_strxor 15 'gnld'
hack
```

CHAPTER 3

Filesystem

This section in general has funcions for file handling.

3.1 bh_bkp

Do quick backup of file using the current system date posfixed to the filename

Note: Usage

`bh_bkp` [filename]

```
$ bh_bkp bashacks.sh
$ ls -1
bashacks.sh
bashacks.sh.20160122
```

3.2 bh_findmime

Find file by mime type

Note: Usage

`bh_findmime` -[type] [directory]

type : -exe, -msi, -txt or -zip

directory : path

```
$ bh_findmime -exe ~/Downloads
/home/bashacks/Downloads//binario.exe

$ bh_findmime -txt ~/Documents
/home/bashacks/Documents//01-text.txt
/home/bashacks/Documents//ha.log

$ bh_findmime -zip ~/
/home/bashacks//crm.zip

$ bh_findmime -msi ~/
/home/bashacks//install.msi
```

3.3 bh_hashes

Generate message digest md5, sha1 and sha256 from file or list of file sent by parameters.

Note: Usage

`bh_hashes` [filename or list of files]

```
$ $ bh_hashes bashacks.sh
5dab37cac730088fd959f8292636fc9b bashacks.sh
38be74a4e710a3eeb24b4fa2015cea990d4eda67 bashacks.sh
587b713bb31e3bf32de0b734805c3dd247f49a14cd9e9a5f35008e4f620d3f82 bashacks.sh
```

3.4 bh_md5rename

Convert filename to equivalent digest md5.

Note: Usage

`bh_md5rename` [filename or list of files]

```
$ touch ment.bin
$ bh_md5rename ment.bin
$ ls
d41d8cd98f00b204e9800998ecf8427e
```

TIP

It's easy compress a file and send it by mail later or protection you from yourself.

3.5 bh_secretfile

A nice feature to any skill, use it to compress one or more files, automatically generating a password and upload to the file.io, in the end of process you'll get a URL and password to decompress file.

Note: Usage

`bh_secretfile` [filename]

```
$ cat > ment.bin
Hi, I'm send this file.
$ bh_secretfile ment.bin
adding: ment.bin (stored 0%)
https://file.io/Raan5CUW8ZTW
NRvC_ZniiEtlwgcrBbI_
```

3.6 bh_sharefile

Just as the bh_secretfile function uploads a file and returns the unique url to access it, this process will not have a password attached, anyone with the URL will be able to download it.

Note: Usage

`bh_sharefile` [filename]

```
$ bh_sharefile texto.txt
https://file.io/EGQvRxqyagIY
```

3.7 bh_zipmal

Copress file in zip format with password protection. the password is `virus`

Note: Usage

`bh_zipmal` [filename]

```
$ bh_zipmal malware.cpl
  adding: malware.cpl (deflated 69%)
-rw-r--r-- 1 bashacks  users  30k Jan 21 23:57 malware.zip
```

CHAPTER 4

Math

In this section, we will have many funcions of mathematics operations that help us in everyday life.

They are not complex to assemble in bash, however, nothing better than a small function to assist, nothing better that send data and get or return result, always read the code.

4.1 bh_bin2dec

This function expects a binary and return its equivalent in decimal.

Note: Usage

`bh_bin2dec` [binary]

binary : string in binary format.

```
$ bh_bin2dec 11111111
255

$ bh_bin2dec 10
2

$ bh_bin2dec 1110
14
```

4.2 bh_charcalc

Think of a way to make operations with 'char', how to sum two positions for a 'char/string' and return letter c or sum of the other and multiply it by 10 and returns 10

Note: Usage

`bh_charcalc` [char/string] [operator] [number]

char/string : string or char to operation operator : * + - number : num of operation

```
$ bh_charcalc A + 2
C

$ bh_charcalc A \* 255
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA.......
```

4.3 bh_dec2bin

Opossed to `bh_bin2dec` this function expects a decimal for converting it into binary.

Note: Usage

`bh_dec2bin` [decimal]

decimal : number in decimal format.

```
$ bh_dec2bin 255
11111111

$ bh_dec2bin 2
10

$ bh_dec2bin 14
1110

#Example

$ for dec in {1..6};
 do
    echo "$dec = $(bh_dec2bin $dec)";
 done

1 = 1
2 = 10
3 = 11
4 = 100
5 = 101
6 = 110
```

4.4 bh_dec2hex

The function expects a input a decimal number it performs the conversion to hex.

Note: Usage

`bh_dec2hex` [decimal]

decimal: number in decimal format

```
$ bh_dec2hex 10
a

$ bh_dec2hex 255
ff
```

4.5 bh_hex2bin

Capture all submitted arguments and convert to binary

Note: Usage

`bh_hex2bin` [list or one hex digit]

```
$ bh_hex2bin abcdef 1 2 3
101010111100110111101111 1 10 11

$ bh_hex2bin 10
10000
```

4.6 bh_hex2dec

This's a conversion function from hex digit to decimal digit

Note: Usage

`bh_hex2dec` [one or more hex digit]

```
$ bh_hex2dec A
10

$ bh_hex2dec FF
255
```

4.7 bh_hexcalc

In the same way as `bh_charcalc`, however, work here with hexdigit.

Note: Usage

`bh_hex2cal` [hex digit] [operator] [hex digit]

```
$ bh_hex2dec A \* 2
0xa0

$ bh_hex2bin FF + 1
0x100
```

CHAPTER 5

Network

5.1 bh_bin2ip

Convert a binary string into network ipaddress.

Note: Usage

`bh_bin2ip` binary string

```
$ bh_bin2ip 00001010.00001010.00000000.00000001
10.10.0.1
```

5.2 bh_hostcalc

Enter a network CIDR mask and know the amount of hosts

Note: Usage

`bh_hostcalc` [mask cidr]

```
$ bh_hostcalc 24
256

$ bh_hostcalc 25
126
```

5.3 bh_ip2bin

Convert network ipaddress into binary string.

Note: Usage

`bh_ip2bin` [ipaddress]

```
$ bh_ip2bin 192.168.0.100
11000000.10101000.00000000.01100100
```

5.4 bh_myip

This returns the external ipaddress of your network connection.

Note: Usage

`bh_myip`

```
$ bh_myip
200.251.1.1
```

5.5 bh_wgetr

Recursive and continue getting a partially-downloaded "if exist" file started by a previous instance of wget with randomize time.

Note: Usage

`bh_wgetr` [url]

```
$ bh_wgetr http://www.mentebinaria.com.br/artigos/0x1e/0x1e-maqengrevwin.html
www.mentebinaria.com.br/art 100%[=========================================>]   8.73K␣
↪ --.-KB/s   in 0s
www.mentebinaria.com.br/rob 100%[=========================================>]     361␣
↪ --.-KB/s   in 0s
www.mentebinaria.com.br/art 100%[=========================================>]  66.18K␣
↪  132KB/s   in 0.5s
$ ls -1
www.mentebinaria.com.br
$ ls -1 www.mentebinaria.com.br/artigos/0x1e/
0x1e-maqengrevwin.html
desktop.png
```

5.6 bh_ipinfo

Query ipinfo.io returns basic info about address.

Note: Usage

`bh_ipinfo` [ipaddress]

```
$ $ bh_ipinfo 8.8.8.8
{
 "ip": "8.8.8.8",
 "hostname": "dns.google",
 "anycast": true,
 "city": "Mountain View",
 "region": "California",
 "country": "US",
 "loc": "37.4056,-122.0775",
 "org": "AS15169 Google LLC",
 "postal": "94043",
 "timezone": "America/Los_Angeles",
 "readme": "https://ipinfo.io/missingauth"
}
```

5.7 bh_unshort

With this function you have the possibility to unshort a URL see below a example.

Note: Usage

`bh_unshort` [URL string]

```
$ bh_unshort http://goo.gl/l6MS
http://googleblog.blogspot.com/2009/12/making-urls-shorter-for-google-toolbar.html
```

5.8 bh_ipisblacklisted

Search for occurrence of the ip address in some blacklist returning [T] if positive and [F] if it is opposite..

Note: Usage

`bh_ipblacklist` [ipaddress]

```
$ bh_ipblacklist 77.xxx.xx.xx
== 77.xxx.xx.xx ==
[F]    TALOS
[F]    Malc0de
[F]    Projecthoneypot.org
[F]    blocklist.de
```

(continues on next page)

(continued from previous page)

```
[T]    Alienvault
[F]    SANS-TOPSOURCE

#if ipaddress is not informed will be considered the outside

$ bh_ipblacklist
== 189.x.xxx.x ==
[F]    TALOS
[F]    Malc0de
[F]    blocklist.de
[F]    Alienvault
[T]    SANS-TOPSOURCE
```

CHAPTER 6

Programming

Session for the creation of facilitators for development in cli

6.1 bh_skel_c

Generates on the standard output `C` skeleton.

Note: Usage

```
bh_skel_c
```

```
$ bh_skel_c
#include <stdio.h>

int main(int argc, char ***argv[]**) {

    return 0;
}
```

6.2 bh_skel_go

Generates on the standard output `go` skeleton.

Note: Usage

```
bh_skel_go
```

```
$ bh_skel_go
package main

import (
    "fmt"
)

func main() {

    fmt.Println("test")
}
```

6.3 bh_skel_latex

Generates on the standard output `latex` skeleton.

Note: Usage

```
bh_skel_latex
```

```
$ bh_skel_latex
\documentclass{article}

\usepackage[english]{babel}
\usepackage[utf8]{inputenc}
\usepackage[margin=1in]{geometry}

\author{}
\title{}

\begin{document}
\maketitle

\end{document}
```

6.4 bh_skel_python

Generates on the standard output `python` skeleton.

Note: Usage

```
bh_skel_python
```

```
$ bh_skel_python
#!/usr/bin/env python
# *-* coding: utf-8 *-*

if __name__ == '__main__':
```

6.5 bh_skel_yara

Generates on the standard output `yara` skeleton.

Note: Usage

```
bh_skel_yara
```

```
$ bh_skel_yara
rule test {
meta:
    author = "mb"
    description = ""
    date = "2022-03-03"
    ref = ""
    hash = ""

strings:
    $a = "test" ascii wide

condition:
    all of them
}
```

CHAPTER 7

Reversing

7.1 bh_asmgrep

With the binary attempts to find instruction asm and print 4 lines around.

Note: Usage

`bh_asmgrep` [asm instruction] [binary path]

```
$ bh_asmgrep mov /bin/ls
....
....
....
--
409f2e:66 90                : xchg   %ax,%ax
409f30:80 7c 13 ff 2f       : cmpb   $0x2f,-0x1(%rbx,%rdx,1)
409f35:48 8d 42 ff          : lea    -0x1(%rdx),%rax
409f39:75 08                : jne    409f43 <__sprintf_chk@plt+0x7783>
409f3b:48 89 c2             : mov    %rax,%rdx
409f3e:48 39 d5             : cmp    %rdx,%rbp
409f41:75 ed                : jne    409f30 <__sprintf_chk@plt+0x7770>
409f43:48 83 c4 08          : add    $0x8,%rsp
--
.....
.....
.....
```

7.2 bh_asminfo

Display information of instructions asm internet is required for help us.

Note: Usage

`bh_asminfo` [asm instruction]

```
$ bh_asminfo mov
mov

|Code   |Mnemonic        |Description                                          |
|88 / r |MOV r/m8, r8    |Move r8 to r/m8                                      |
|89 / r |MOV r/m16, r16  |Move r16 to r/m16                                    |
|89 / r |MOV r/m32, r32  |Move r32 to r/m32                                    |
|8A / r |MOV r8, r/m8    |Move r/m8 to r8                                      |
|8B / r |MOV r16, r/m16  |Move r/m16 to r16                                    |
.......
.......
.......
```

7.3 bh_replacestring

Find and replace string occurrence in the file, attention: the original file will be replacede by the new generated file.

Note: Usage

`bh_replacestring` [file] [string to search] [string to replace]

```
$ hexdump -C MB_DEV
.......
00000690  2e 00 54 00 58 00 54 00  2e 00 00 00 73 00 77 00  |..T.X.T.....s.w.|
000006a0  e5 45 53 54 45 54 7e 31  53 57 58 20 00 65 a1 9b  |.ESTET~1SWX .e..|
000006b0  8b 54 8b 54 00 00 a1 9b  8b 54 00 00 00 00 00 00  |.T.T.....T......|
000006c0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00005e00  4d 65 6e 74 65 42 69 6e  61 72 69 61 0a 00 00 00  |MenteBinaria....|
00005e10  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
$ bh_replacestring MB_DEV MenteBinaria BinariaMente
$ hexdump -C MB_DEV
.......
00000690  2e 00 54 00 58 00 54 00  2e 00 00 00 73 00 77 00  |..T.X.T.....s.w.|
000006a0  e5 45 53 54 45 54 7e 31  53 57 58 20 00 65 a1 9b  |.ESTET~1SWX .e..|
000006b0  8b 54 8b 54 00 00 a1 9b  8b 54 00 00 00 00 00 00  |.T.T.....T......|
000006c0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00005e00  42 69 6e 61 72 69 61 4d  65 6e 74 65 0a 00 00 00  |BinariaMente....|
00005e10  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
```

7.4 bh_zerostring

Replace with zero bytes in block or common file.

Note: Usage

`bh_zerostring` [file] [string to replace]

```
# hexdump -C MB_DEV
.....
00005860  41 4d 00 42 00 2d 00 66  00 69 00 0f 00 a1 6c 00  |AM.B.-.f.i....l.|
00005870  65 00 2e 00 74 00 78 00  74 00 00 00 00 00 ff ff  |e...t.x.t.......|
00005880  4d 42 2d 46 49 4c 45 20  54 58 54 20 00 41 26 be  |MB-FILE TXT .A&.|
00005890  69 54 69 54 00 00 26 be  69 54 05 00 1b 00 00 00  |iTiT..&.iT......|
.....
# bh_zerostring MB_DEV MB-FILE
7+0 records in
7+0 records out
7 bytes copied, 7.3484e-05 s, 95.3 kB/s
# hexdump -C MB_DEV
.....
00005860  41 4d 00 42 00 2d 00 66  00 69 00 0f 00 a1 6c 00  |AM.B.-.f.i....l.|
00005870  65 00 2e 00 74 00 78 00  74 00 00 00 00 00 ff ff  |e...t.x.t.......|
00005880  00 00 00 00 00 00 00 20  54 58 54 20 00 41 26 be  |....... TXT .A&.|
00005890  69 54 69 54 00 00 26 be  69 54 05 00 1b 00 00 00  |iTiT..&.iT......|
.....
# mount -o loop -t vfat MB_DEV /mnt/
# ls -la /mnt/
total 16
drwxr-xr-x 2 root root 16384 Dec 31  1969 .
drwxr-xr-x 1 root root   152 Feb 17 15:21 ..
```

CHAPTER 8

String

In this section, you are see funcions for string manipulate.

8.1 bh_asc2dec

This function performs the conversion of a char on it's decimal equivalent.

Note: `bh_asc2dec` [char]

```
$  bh_asc2dec a
97
$ bh_asc2dec A
65
```

8.2 bh_asciitable

Display in the terminal ascii table, if you are a programmer knows how this's important.

Note: `bh_asciitable`

```
$ bh_asciitable

Dec Hex    Dec Hex    Dec Hex  Dec Hex  Dec Hex  Dec Hex   Dec Hex   Dec Hex
  0 00 NUL  16 10 DLE  32 20    48 30 0  64 40 @  80 50 P   96 60 `  112 70 p
  1 01 SOH  17 11 DC1  33 21 !  49 31 1  65 41 A  81 51 Q   97 61 a  113 71 q
  2 02 STX  18 12 DC2  34 22 "  50 32 2  66 42 B  82 52 R   98 62 b  114 72 r
  3 03 ETX  19 13 DC3  35 23 #  51 33 3  67 43 C  83 53 S   99 63 c  115 73 s
```

(continues on next page)

(continued from previous page)

```
  4 04 EOT   20 14 DC4   36 24 $   52 34 4   68 44 D   84 54 T   100 64 d   116 74 t
  5 05 ENQ   21 15 NAK   37 25 %   53 35 5   69 45 E   85 55 U   101 65 e   117 75 u
  6 06 ACK   22 16 SYN   38 26 &   54 36 6   70 46 F   86 56 V   102 66 f   118 76 v
  7 07 BEL   23 17 ETB   39 27 '   55 37 7   71 47 G   87 57 W   103 67 g   119 77 w
  8 08 BS    24 18 CAN   40 28 (   56 38 8   72 48 H   88 58 X   104 68 h   120 78 x
  9 09 HT    25 19 EM    41 29 )   57 39 9   73 49 I   89 59 Y   105 69 i   121 79 y
 10 0A LF    26 1A SUB   42 2A *   58 3A :   74 4A J   90 5A Z   106 6A j   122 7A z
 11 0B VT    27 1B ESC   43 2B +   59 3B ;   75 4B K   91 5B [   107 6B k   123 7B {
 12 0C FF    28 1C FS    44 2C ,   60 3C <   76 4C L   92 5C \   108 6C l   124 7C |
 13 0D CR    29 1D GS    45 2D -   61 3D =   77 4D M   93 5D ]   109 6D m   125 7D }
 14 0E SO    30 1E RS    46 2E .   62 3E >   78 4E N   94 5E ^   110 6E n   126 7E ~
 15 0F SI    31 1F US    47 2F /   63 3F ?   79 4F O   95 5F _   111 6F o   127 7F DEL
```

8.3 bh_dec2asc

Once having to enter a decimal returns it's equivalent in ascii.

Note: `bh_dec2asc` [decimal]

decimal : dec equivalent of the ascii table to convert

```
$ bh_dec2asc 65
A
$ bh_dec2asc 41
)
```

8.4 bh_hex2str

Converts one or more bytes into a hex string to str.

Accepts as input all output formats str2hex function.

Note: `bh_hex2str` [hex string]

```
$ bh_hex2str '72 6f 63 6b'
rock
$ bh_hex2str '0x726f636b'
rock
$ bh_hex2str '0x72 0x6f 0x63 0x6b'
rock
$ bh_hex2str '{0x72, 0x6f, 0x63, 0x6b}'
rock
```

8.5 bh_str2dec

Convert one or more bytes to their decimal equivalent.

Note: `bh_str2dec` [char or string]

```
$ bh_str2dec A
65
$ bh_str2dec mbin
109 98 105 110
$ bh_str2dec root
114 111 111 116
```

8.6 bh_str2hexr

Converts string in hex byte equivalent to each char (hex string). reverse mode

Note: `bh_str2hexr` [-x] [-0x] [-c] [string]

```
$ bh_str2hexr 'Fernando'
6f 64 6e 61 6e 72 65 46
$ bh_str2hexr -x 'Fernando'
\x6f\x64\x6e\x61\x6e\x72\x65\x46
$ bh_str2hexr -0x 'Fernado'
0x6f 0x64 0x6e 0x61 0x6e 0x72 0x65 0x46
```

8.7 bh_str2hex

Converts string in hex byte equivalent to each char (hex string).

Note: `bh_str2hex` [-x] [-0x] [-c] [string]

```
$ bh_str2hex 'Fernando'
46 65 72 6e 61 6e 64 6f
$ bh_str2hex -x 'Fernando'
\x46\x65\x72\x6e\x61\x6e\x64\x6f
$ bh_str2hex -0x 'Fernado'
0x46 0x65 0x72 0x6e 0x61 0x6e 0x64 0x6f
```

8.8 bh_urldecode

Decode string with bh_urldecode from web standard to human format.

Note: `bh_urldecode` [encoded string]

```
$ bh_urldecode '%2fzzz%21%40%2e%23'
/zzz!@.#
```

8.9 bh_urlencode

Encoded string with bh_urlencode to web standard.

Note: `bh_urlencode` [string]

```
$ bh_urlencode '/zzz!@.#'
%2fzzz%21%40%2e%23
```

8.10 bh_utf8table

Show UTF8 table.

Note: `bh_utf8table`

```
$ bh_utf8table
Hex       Hex       Hex       Hex       Hex       Hex       Hex       Hex
c2 a0     c2 ac ¬   c2 b8 ¸   c3 84 Ä   c3 90 Ð   c3 9c Ü   c3 a8 è   c3 b4 ô
c2 a1 ¡   c2 ad      c2 b9 ¹   c3 85 Å   c3 91 Ñ   c3 9d Ý   c3 a9 é   c3 b5 õ
c2 a2 ¢   c2 ae ®   c2 ba º   c3 86 Æ   c3 92 Ò   c3 9e Þ   c3 aa ê   c3 b6 ö
c2 a3 £   c2 af ¯   c2 bb »   c3 87 Ç   c3 93 Ó   c3 9f ß   c3 ab ë   c3 b7 ÷
c2 a4 ¤   c2 b0 °   c2 bc ¼   c3 88 È   c3 94 Ô   c3 a0 à   c3 ac ì   c3 b8 ø
c2 a5 ¥   c2 b1 ±   c2 bd ½   c3 89 É   c3 95 Õ   c3 a1 á   c3 ad í   c3 b9 ù
c2 a6 ¦   c2 b2 ²   c2 be ¾   c3 8a Ê   c3 96 Ö   c3 a2 â   c3 ae î   c3 ba ú
c2 a7 §   c2 b3 ³   c2 bf ¿   c3 8b Ë   c3 97 ×   c3 a3 ã   c3 af ï   c3 bb û
c2 a8 ¨   c2 b4 ´   c3 80 À   c3 8c Ì   c3 98 Ø   c3 a4 ä   c3 b0 ð   c3 bc ü
c2 a9 ©   c2 b5 µ   c3 81 Á   c3 8d Í   c3 99 Ù   c3 a5 å   c3 b1 ñ   c3 bd ý
c2 aa ª   c2 b6 ¶   c3 82 Â   c3 8e Î   c3 9a Ú   c3 a6 æ   c3 b2 ò   c3 be þ
c2 ab «   c2 b7 ·   c3 83 Ã   c3 8f Ï   c3 9b Û   c3 a7 ç   c3 b3 ó   c3 bf ÿ
```