Machine Translated by Google

Unicode support

Unicode is a table of the ratio of characters of various languages of the world with a fixed number or code point. This section covers Unicode support in Cobalt Strike.

Encodings

Unicode is a table of relationships between characters and numbers (code points), but it is not an encoding. An encoding is a consistent way of giving meaning to individual characters or sequences of bytes by matching them to code points in this table.

Java applications use UTF-16 encoding to store and manipulate characters. UTF-16 is an encoding that uses two bytes to represent normal characters. Rarer characters are represented by four bytes. Cobalt Strike is a Java application and is capable of storing, managing and displaying text in various world languages. At the core of the Java platform, there are no major technical barriers to
this.

In a Windows environment, things are a little different. The character representation capabilities of Windows go back to the days of DOS. DOS programs work with ASCII text and nice rectangle drawing characters. The common encoding for displaying the numbers 0-127 in US ASCII and 128-255 in those lovely box-drawing characters has a name. This is code page 437. There are several variants of code page 437 that combine the beautiful box-drawing characters with characters from specific languages. This encoding set is known as the OEM encoding. Today, every instance of Windows has a global OEM encoding setting. This setting determines how to interpret the output of bytes written by the program to the console. To correctly interpret cmd.exe output, it is important to know the target's OEM encoding.

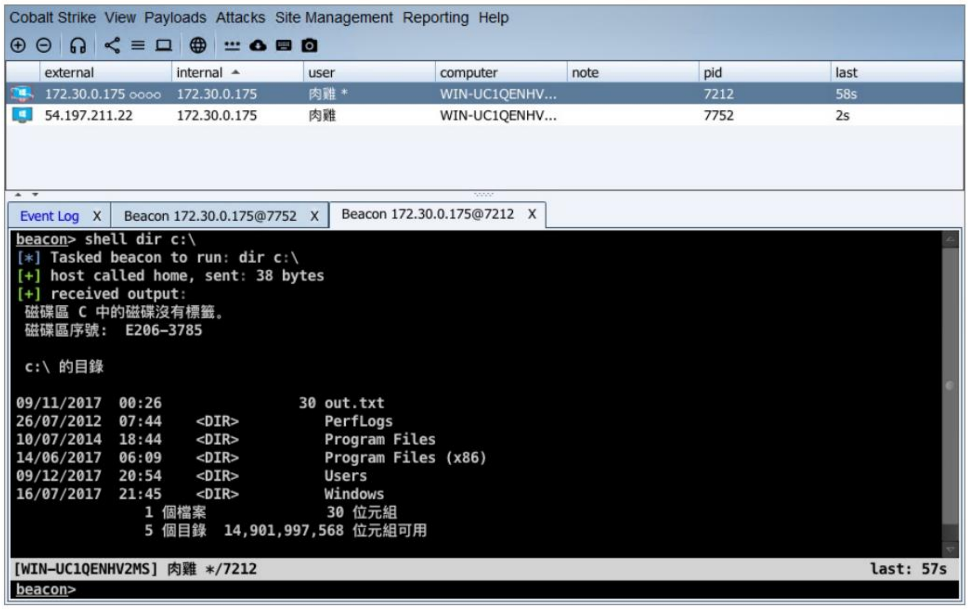
However, the fun continues. Rectangle-drawing symbols are required by DOS programs, but not at all required by Windows programs. Therefore, Windows has the concept of ANSI encoding. This is a global setting, similar to OEM encoding. The ANSI encoding dictates how the ANSI Win32 APIs map sequences of bytes to code points. The ANSI encoding of a language allows you to abandon the characters for drawing rectangles in favor of those characters that are necessary for the language for which the encoding is designed. An encoding may not be limited to mapping one byte to one character. A variable length encoding can represent the most common characters as a single byte and the rest as some sequence of multiple bytes.

However, ANSI encodings are far from the whole story. Windows APIs often have both ANSI and Unicode variants. The ANSI version of the API accepts and interprets a text argument as described above. The Unicode Win32 API expects UTF-16 encoded text arguments.

On Windows, various encoding options are possible. There is an OEM encoding that can represent specific text in a customized language. There is an ANSI encoding that can represent more text, mostly in a given language. And there is UTF-16 which can contain any code point. There is also UTF-8 encoding, which is a variable length encoding that takes up less space for ASCII text, but can also contain any code point.

Beacon

Beacon transmits the target's ANSI and OEM encodings as part of the session metadata. Cobalt Strike uses these values to encode input text into the target's encoding, if necessary. Cobalt Strike also uses these values to decode output text if needed.


Cobalt Strike View Payloads Attacks Site Management Reporting Help
Event Log X
Beacon 172.30.0.175@7752 X
Beacon 172.30.0.175@7212 X

external	internal	user	computer	note	pid	last
172.30.0.175 ∞∞	172.30.0.175	肉雞 *	WIN-UC1QENHV...		7212	58s
54.197.211.22	172.30.0.175	肉雞	WIN-UC1QENHV...		7752	2s

```
beacon> shell dir c:\
[*] Tasked beacon to run: dir c:\
[+] host called home, sent: 38 bytes
[+] received output:
 磁碟區 C 中的磁碟沒有標籤。
 磁碟區序號:  E206-3785

 c:\ 的目錄

09/11/2017  00:26                30 out.txt
26/07/2012  07:44    <DIR>          PerfLogs
10/07/2014  18:44    <DIR>          Program Files
14/06/2017  06:09    <DIR>          Program Files (x86)
09/12/2017  20:54    <DIR>          Users
16/07/2017  21:45    <DIR>          Windows
               1 個檔案              30 位元組
               5 個目錄  14,901,997,568 位元組可用
[WIN-UC1QENHV2MS] 肉雞 */7212                                        last: 57s
beacon>
```

In general, translating text to and from the target encoding is a transparent process. If you're working with a target that's set to a specific language, everything will work as you'd expect. In the case of working with mixed

language environments, there will be differences in the behavior of the commands. For example, if the output contains Cyrillic, Chinese, and Latin characters, some commands will work correctly. Others don't. Most Beacon commands use the target's ANSI encoding to encode

input and decode output. The configured ANSI encoding can only map characters to code points for a small number of languages. If the current target's ANSI encoding does not display Cyrillic characters, make_token will not work correctly with a username or password that uses Cyrillic characters.

Some Beacon commands use UTF-8 for input and output. Such commands generally do what you would expect from multilingual content. This is because UTF-8 text can match characters with any Unicode code point.

The following table shows which Beacon commands use non-ANSI encoding to decode input and output:

Team	Input encoding	Output encoding
hashdump		UTF-8
mimikatz	UTF-8	UTF-8

Machine Translated by Google

Team	Input encoding	Output encoding
powerpick	UTF-8	UTF-8
powershell	UTF-16	OEM
psinject	UTF-8	UTF-8
shell	ANSI	OEM

NOTE:
For those familiar with mimikatz, you'll notice that mimikatz internally uses the Unicode Win32 API and UTF-16 characters. Where does UTF-8 come from? Cobalt Strike's mimikatz interface sends input as UTF-8 and converts output to UTF-8.

SSH sessions

Cobalt Strike's SSH sessions use UTF-8 encoding for input and output.

Logging

Cobalt Strike's logs are UTF-8 encoded text.

Fonts

Your font may not support displaying characters in some languages. To change Cobalt Strike's fonts: Go to **Cobalt Strike**

-> Preferences -> Cobalt Strike to change the GUI Font value. This will change the font that Cobalt Strike uses in dialog boxes, tables, and the rest of the interface. Go to **Cobalt Strike -> Preferences -> Console** to change the

font used in consoles. In **Cobalt Strike -> Preferences -> Graph** there is a Font option to change the font used

by the Pivot graph.