Fuzzing glibc’s *iconv* program

Arjun Shankar
arjun@redhat.com
iconv

- Program
  
  ```
  $ iconv -f ISO-8859-2 -t UTF-8
  ```

- Function
  
  ```
  cd = iconv_open ("UTF-8", "ISO-8859-2");
  ir = iconv (cd, ibuf, isz, obuf, osz);
  ```
Transliteration and Invalid Input

- Basically, á is almost the same as ‘a’:
  
  ```
  $ echo áa | iconv -f UTF-8 -t ASCII//TRANSLIT
  aa
  ```

- Skip invalid/non-representable chars:
  
  ```
  echo áa | iconv -f UTF-8 -t ASCII//IGNORE
  a
  ```

  `iconv: illegal input sequence at position 4`
Bug 19519 (2016)

Sometimes, `iconv` hangs:

```
$ echo -en '\x80' \\
  | iconv -f ASCII -t ASCII//TRANSLIT//IGNORE -c
```

- Order of suffixes seemed important
  ```
  -f ASCII -t ASCII//IGNORE//TRANSLIT -c
  ```
- Code manipulates suffix strings
Working with limited knowledge

- Don’t really know much about `iconv`
- `0x80` isn’t a valid ASCII character
- Code doesn’t handle suffixes cleanly
- What other character sets are affected, & is the problem really just suffixes?

=> Let a program figure it out!
Fuzzer

- Bash script
  ```bash
  echo $twobyte \n  | iconv $c -f $charset -t "UTF-8$suffixes"
  ```
- For all combinations of two bytes, for all character sets, for all reasonable combinations of suffixes
- Make 10 cups of coffee?
Results

- 167 charsets had hangs, 162 with `//TRANSLIT//IGNORE`
- 5 were converter bugs
- Decided to target the suffix handling
- Replaced string manipulation with conversion specification struct
What next?

- Converter hangs
- //IGNORE can mean different things for input and output
- New interface? New program options?
- Bug 26383: bind_textdomain_codeset
- Improve test coverage