index

- Internationalization Security
- Check for Spam and Other Malicious Strings
- Escape Internationalized Strings
- Use Placeholders for URLs
- Compile Your Own .mo Binaries

Internationalization Security

Security is often overlooked when talking about internationalization, but there are a few important things to keep in mind.

Check for Spam and Other Malicious Strings

When a translator submits a localization to you, always check to make sure they didn't include spam or other malicious words in their translation. You can use Google Translate to translate their translation back into your native language so that you can easily compare the original and translated strings.

Escape Internationalized Strings

You can't trust that a translator will only add benign text to their localization; if they want to, they could add malicious JavaScript or other code instead. To protect against that, it's important to treat internationalized strings like you would any other untrusted input.

If you're outputting the strings, then they should be escaped.

Insecure

```
_e( 'The REST API content endpoints were added in WordPress 4.7.', 'your-text-domain' );
```

Secure

```
esc_html_e( 'The REST API content endpoints were added in WordPress 4.7.', 'your-text-domain'
);
```

Alternatively, some people choose to rely on a translation verification mechanism, rather than adding escaping to their code. One example of a verification mechanism is the editor roles that the WordPress Polyglots team uses for translate.wordpress.org. This ensures that any translation submitted by an untrusted contributor has been verified by a trusted editor before being accepted.

Use Placeholders for URLs

Don't include URLs in internationalized strings, because a malicious translator could change them to point to a different URL. Instead, use placeholders for `printf()` or `sprintf()`.

Insecure

```
_e(
  'Please <a href="https://login.wordpress.org/register"> register for a WordPress.org
account</a>.',
  'your-text-domain'
);
```

Secure

```
printf(
  esc_html__( 'Please %1$s register for a WordPress.org account %2$s.', 'your-text-domain' ),
  '<a href="https://login.wordpress.org/register">',
  '</a>'
);
```

Compile Your Own .mo Binaries

Often translators will send the compiled .mo file along with the plaintext .po file, but you should discard their .mo file and compile your own, because you have no way of knowing whether or not it was compiled from the corresponding .po file, or a different one. If it was compiled against a different one, then it could contain spam and other malicious strings without your knowledge.

Using PoEdit to generate the binary will override the headers in the .po file, so instead it's better to compile it from the command line:

```
msgfmt -cv -o /path/to/output.mo /path/to/input.po
```

□□□ #2
□ □□□ □□□ 2026-10-26 15:37:07 UTC
□ □□□ □□□ 2026-10-26 15:37:07 UTC