



Zip Slip is a widespread arbitrary file overwrite vulnerability, which typically results in remote command execution. It is exploited by extracting archives which contain directory traversal filenames that are not validated by user or library code.

**Vulnerable code:**

```
Enumeration<ZipEntry> entries = zip.getEntries();
while (entries.hasMoreElements()) {
    ZipEntry e = entries.nextElement();
    File f = new File(destinationDir, e.getName());
    InputStream input = zip.getInputStream(e);
    IOUtils.copy(input, write(f));
}
```

Validation code:

```
String canonicalDestDirPath = destDir.getCanonicalPath();
File destfile = new File(destDir, e.getName());
String canonicalDestFile destfile.getCanonicalPath();
if (!canonicalDestFile.startsWith(canonicalDestDirPath +
    File.separator)) {
    throw new ArchiverException("Entry outside of the target dir");
}
```

**Vulnerable code:**

```
self.on('entry', function(entry) {
    entry.pipe(Writer({
        path: path.join(opts.path, entry.path)
    })))
}))
```

Validation code:

```
var filePath = path.join(targetFolder, entry.path);
if (filePath.indexOf(path.join(targetFolder, path.sep)) != 0) {
    return;
}
```

**Go****Vulnerable code:**

```
func (rarFormat) Read(input io.Reader, dest string) {
    rr := rardecode.NewReader(input, "")
    for {
        header := rr.Next()
        writeNewFile(filepath.Join(dest, header.Name), rr,
            header.Mode())
    }
}
```

Validation code:

```
func sanitizePath(filePath string, dest string) error {
    destpath := filepath.Join(dest, filePath)
    if !strings.HasPrefix(destpath, filepath.Clean(dest) +
        string(os.PathSeparator)) {
        return fmt.Errorf("%s: illegal file path", filePath)
    }
    return nil
}
```

**Vulnerable code:**

```
public static void WriteToDirectory(IArchiveEntry entry,
    string destDirectory,
    ExtractionOptions options) {
    string file = Path.GetFileName(entry.Key);
    string destFileName = Path.Combine(destDirectory, file);
    entry.WriteToFile(destFileName, options);
}
```

Validation code:

```
destFile = Path.GetFullPath(Path.Combine(destDir, entry.Key));
string fullDestDir= Path.GetFullPath (destDir +
    Path.DirectorySeparatorChar);
if (!destFileName.StartsWith(fullDestDir)) {
    throw new ExtractException("Entry outside of target dir");
}
```

Vulnerable Libraries

| Library | Fixed Version | CVE |
|--|---------------|------------------|
| codehaus/plexus-archiver | 3.6.0 | CVE-2018-1002200 |
| zeroturnaround/zt-zip | 1.13 | CVE-2018-1002201 |
| zip4j | 1.3.3 | CVE-2018-1002202 |
| unzipper | 0.8.13 | CVE-2018-1002203 |
| npm library | 0.4.9 | CVE-2018-1002204 |
| DotNetZip.Semverd | 1.11.0 | CVE-2018-1002205 |
| SharpCompress | 0.21.0 | CVE-2018-1002206 |
| mholt/archiver | e4ef56d4 | CVE-2018-1002207 |
| quazip | 0.7.6 | CVE-2018-1002209 |
| SharpZipLib | Unmaintained | CVE-2018-1002208 |

For the latest info on these libraries and those from other ecosystems, check out <http://github.com/snyk/zip-slip-vulnerability>

For more details about Zip Slip and further information about other ecosystems, visit the [Snyk research pages](#)

