- Nel caso fosse necessario utilizzare dati utente non attendibili per selezionare la libreria da caricare, verificare che l'input corrisponda a un insieme predefinito di nomi rigidamente indicati in una "white list" o comunque selezionare esclusivamente da elenchi di nomi controllati relativamente a possibili librerie software.

Esempio
Forma non corretta (con lettura dinamica di una libreria indicate in modo arbitrario da un utente):

```
var qs = require('querystring');
var server = http.createServer(function (request, response) {
      var libName = qs.parse(request.url).libName;
      if (typeof libName != "undefined") {
            var dynamicLib = require(libName);
  }
}
```

Forma corretta tramite “white list”:

```
var qs = require('querystring');
var server = http.createServer(function (request, response) {
      var libName = qs.parse(request.url).libName;
      var dynamicLib;
      if (typeof libName != "undefined") {
            if (libName == 'user')
            dynamicLib = require('userLib');
      else if (libName == 'special')
      dynamicLib = require('specialUserLib');
      else
      dynamicLib = require('anonymousLib');
  }
}
```

Per ulteriori informazioni si veda: http://cwe.mitre.org/data/definitions/98.html,
CWE-98: Improper Control of Filename for Include/Require Statement in PHP Program ('PHP Remote File Inclusion').

7.9.6 File Manipulation

Come riconoscerla

Se un malintenzionato può influire su file arbitrari di propria scelta ed è in grado di sovrascrivere o corrompere file di sistema, Potrebbe agevolmente causare un DoS (denial of service). Se il malintenzionato in questione ha la possibilità di modificare il contenuto di detti file, il pericolo che venga eseguito del codice dannoso è molto concreta.
Questa vulnerabilità (indicata con il nome esteso di “Files or Directories Accessible to External Parties”) ha come conseguenza la possibilità che file o directory siano accessibili ad utenti esterni malintenzionati.
È una variante della vulnerabilità indicata come File Disclosure con possibile manipolazione di file di sistema esistenti sul server attaccato.

Come difendersi

Prendere in considerazione l'utilizzo di una soluzione statica per i file a cui è consentita la scrittura. Ad esempio un elenco di file scrivibili verificati o una diversa soluzione di archiviazione dei file, come un database. Se assolutamente necessario, limitare la scrittura della destinazione in una singola cartella disinfettando correttamente gli input forniti dall'utente per il nome di file e cartelle. Considerare di integrare questo con un segno di spunta per garantire l'esistenza o meno di un file, in base ai requisiti aziendali del codice dell'applicazione.

Esempio:
Codice vulnerabile:

```
if (isset($_GET['lognamè]) && isset($_GET['action'])) {
```