Detecting Node.js prototype pollution vulnerabilities via object lookup analysis

Abstract

Prototype pollution is a type of vulnerability specific to prototype-based languages, such as JavaScript, which allows an adversary to pollute a base object's property, leading to a further consequence such as Denial of Service (DoS), arbitrary code execution, and session fixation. On one hand, the only prior work in detecting prototype pollution adopts dynamic analysis to fuzz package inputs, which inevitably has code coverage issues in triggering some deeply embedded vulnerabilities. On the other hand, it is challenging to apply state-of-the-art static analysis in detecting prototype pollution because of the involvement of prototype chains and fine-grained object relations including built-in ones.

In this paper, we propose a flow-, context-, and branch-sensitive static taint analysis tool, called ObjLupAnsys, to detect prototype pollution vulnerabilities. The key of ObjLupAnsys is a so-called object lookup analysis, which gradually expands the source and sink objects into big clusters with a complex inner structure by performing targeted object lookups in both clusters so that a system built-in function can be redefined. Specifically, at the source cluster, ObjLupAnsys proactively creates new object properties based on how the target program uses the initial source object; at the sink cluster, ObjLupAnsys assigns property values in object lookups to decrease the number of object lookups to reach a system built-in function.

We implemented an open-source tool and applied it for the detection of prototype pollution among Node.js packages. Our evaluation shows that ObjLupAnsys finds 61 zero-day, previously-unknown, exploitable vulnerabilities as opposed to 18 by the state-of-the-art dynamic fuzzing tool and three by a state-of-the-art static analysis tool that is modified to detect prototype pollution. To date, 11 vulnerable Node.js packages are assigned with CVE numbers and five have already been patched by their developers. In addition, ObjLupAnsys also discovered seven applications or packages including a real-world, online website, which are indirectly vulnerable due to the inclusion of vulnerable packages found by ObjLupAnsys.

Bibliography

Song Li, Mingqing Kang, Jianwei Hou, and Yinzhi Cao. Detecting node.js prototype pollution vulnerabilities via object lookup analysis. In *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2021, page 268–279, New York, NY, USA, 2021. Association for Computing Machinery.

Link and DOI

Link: https://dl.acm.org/doi/10.1145/3468264.3468542
DOI: 10.5281/zenodo.5092777

Detecting Node.js Prototype Pollution Vulnerabilities via Object Lookup Analysis

Song Li, Mingqing Kang, Jianwei Hou, Yinzhi Cao

Johns Hopkins University | Whiting School of Engineering | Baltimore, MD

The paper was published to ESEC/FSE 2021

Introduction

We design and develop a **flow-, context-, and branch-sensitive static taint analysis tool**, called ObjLupAnsys, to detect **prototype pollution** vulnerabilities in Node.JS packages

Our contributions

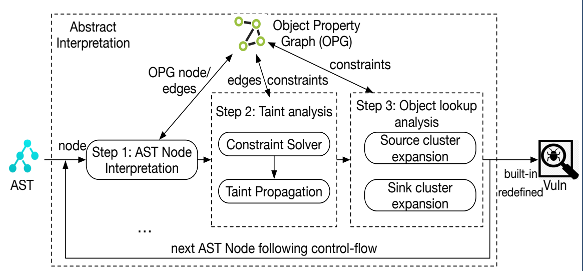
1) We design a novel object lookup analysis and proposed a graph structure, called **Object Property Graph (OPG)**, to support such an analysis in detecting prototype pollution vulnerabilities
2) We implement an **open-source** framework, called **ObjLupAnsys**, to generate OPG, perform object lookup analysis, and detect prototype pollutions
3) ObjLupAnsys found **61** exploitable zero-day vulnerabilities and also detected **seven** indirectly-vulnerable ones due to inclusion of vulnerable packages (11 of them being assigned with CVE identifiers so far).

Motivating Example

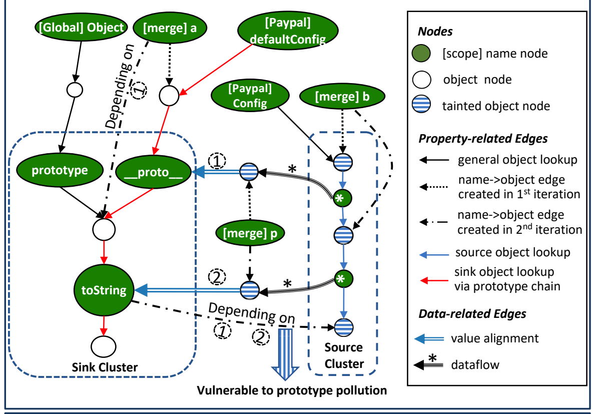
```
function merge(a, b) {
  for (var p in b) {
    try {
        if (b[p].constructor === Object)
          a[p] = merge(a[p], b[p]);
        else
          a[p] = b[p];
    } catch (e) {
        a[p] = b[p];
    }
  }
  return a;
}
...
var Paypal = function (config) {
  if (!config.userId)
   throw new Error('Config must have userId');
  ...
  this.config = merge(defaultConfig, config);
};
...
module.exports = Paypal;
// Expolit ====================================
var PayPal = require('paypal-adaptive');
var p = new PayPal(JSON.parse(
    '{"__proto__": {"toString": "polluted"}, "userId":
    "foo", "password": "bar", "signature": "abcd",
    "appId": "1234", "sandbox": "1234"}'))
console.log(({}).toString);
```

Methods

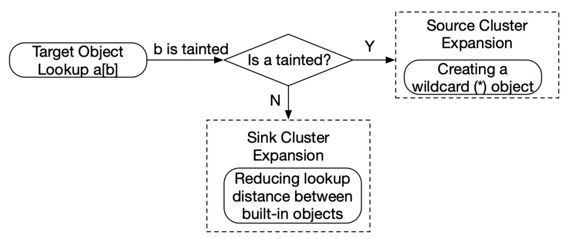
System Architecture



Simplified Object Property Graph for the Motivating Example



Flowchart for Object Lookup Analysis


Target Object Lookup a[b]
b is tainted
Is a tainted?
Y
Source Cluster Expansion
Creating a wildcard (*) object
N
Sink Cluster Expansion
Reducing lookup distance between built-in objects

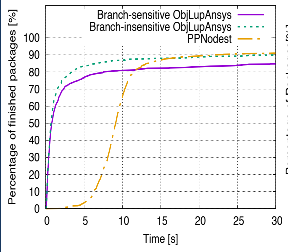
Results

TP/FP/FN of ObjLupAnsys and the existing tools

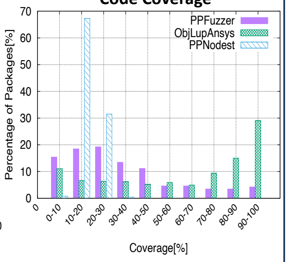
Name	Real-world NPM Packages		Legacy CVE Packages (52 packages in total)	
	TP	FP	TP	FN
PPFuzzer	18	0	32	20
PPNodest	3	3	6	46
ObjLupAnsys (branch-insensitive)	38	14	28	24
ObjLupAnsys (branch-sensitive)	**61 (confirmed) (11 CVEs)**	**20**	**40**	**12**

False Positive Rate: 20 / (61 + 20) = 24.6%
False Negative Rate: 12 / 52 = 23.0%

Performance



Code Coverage



Finishing 90%/85% (branch-insensitive/-sensitive) of the packages in 30s
Median code coverage: 71.9%. (28.0% for PPFuzzer and 19.0% for PPNodest)

Indirectly-vulnerable Applications/Packages

Vulnerable Package	Indirectly-vulnerable Applications/Packages
undefsafe	http://jsonbin.org
dset	design-system-utils (1.5.0), weoptions (0.0.11), quaff (4.2.0)
just-safe-set	magasin (0.2.2)
object-set	node-architect (0.0.15)
simple-odata-server	the default server for the package

A selective list of zero-day vulnerabilities found by ObjLupAnsys

Vulnerable Package	Weekly Download	Vulnerable Package	Weekly Download
undefsafe	2,532,740	dot-object	109,419
append-field	1,301,874	fastest-validator	28,811
graphql-anywhere	386,530	eivindfjeldstad-dot	11,511
aws-xray-sdk-core	187,901	mathjax-full	4,621
cli-table-redemption	178,822	paypal-adaptive	1,890