// BEERUMP 2026

Extension malveillante, zéro fichier malveillant. Normal.

Ghost Anchor Persistence · Chromium-based browsers vulnerability · Fir3n0x

Fir3n0x

Offensive Security / Red Team

// FIND ME

- x.com/fir3n0x_
- github.com/Fir3n0x
- linkedin.com/in/corentin-mahieu
- fir3n0x.github.io

// CONTEXTE DE RECHERCHE

Extensions malveillantes comme agents C2

Le navigateur comme vecteur d'implant

Plutôt que de compromettre le système, on cible l'environnement navigateur — nativement de confiance, peu surveillé par les SOC.

Une extension = un agent C2 discret

Accès natif aux cookies, au réseau, aux pages web, aux credentials stockés. Aucun processus suspect, aucun binaire déposé.

`post-exploitation · accès initial · pas de priv. admin requis`

Question de recherche

Peut-on déployer un agent extension de façon persistante et furtive — sans laisser de trace détectable sur le système de fichiers ?

// ANATOMIE D'UNE EXTENSION CHROMIUM

Structure d'une extension

{} **manifest.json**

Fichier de config obligatoire — déclare l'identité, les permissions, les scripts et ressources de l'extension.

JS **Content Scripts**

Fichiers JS injectés dans les pages visitées — accès complet au DOM, lecture et modification du contenu affiché.

SW **Service Worker / Background Script**

Script JS exécuté en arrière-plan, accès aux APIs privilégiées (cookies, webRequest, scripting). Cible principale d'un attaquant.

```
{
  "manifest_version": 3,
  "name": "Test Communication Serveur",
  "version": "1.0",
  "description": "Communication avec serveur python",
  "permissions": [
    "alarms"
  ],
  "host_permissions": [
    "https://domain.com/"
  ],
  "background": {
    "service_worker": "background.js"
  },
  "content_scripts": [
    {
      "matches": [
        "<all_urls>"
      ],
      "js": [
        "content.js"
      ]
    }
  ],
  "key": "MIIBIjANBgkqhkiG9w0[...]Sej5JnUDpU5xoCY+wN0+Lu/1K/y/
0zheb24qSSQQ4qm5yDn1g/V8CC42eL/T0Xx2EfMNKb4dHpOs0sCmy2TqLbCp/
cyFYfaDdwIDAQAB"
}
```

// CE QU'ON PEUT FAIRE AVEC UNE EXTENSION AGENT

Spoiler : beaucoup trop de choses

Session hijacking

Exfiltration de cookies — accès direct aux comptes sans mot de passe

`chrome.cookies.getAll()`

Keylogger + presse-papiers

Capture de tout ce que l'utilisateur tape ou copie dans le navigateur

`scripting.executeScript()`

Exfil conversations LLM

Lecture des échanges ChatGPT, Claude, Gemini depuis le DOM en temps réel

`DOM observer`

Lecture de fichiers

Accès arbitraire au système de fichiers via les APIs navigateur

`file system API`

Proxy réseau

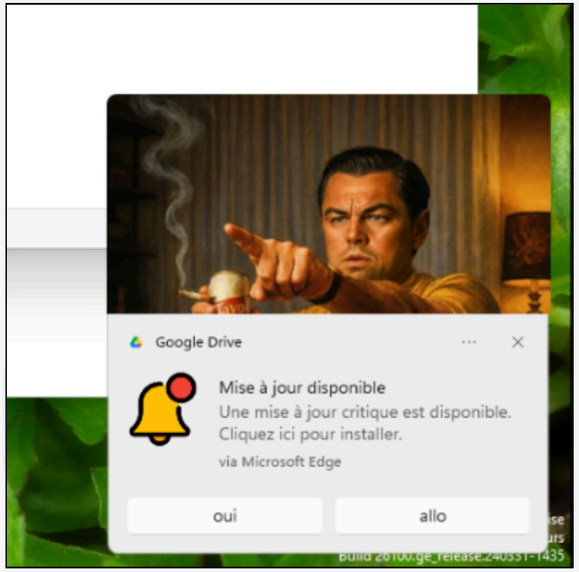
Interception et redirection du trafic navigateur

`proxy.settings`

Injection JS dans le DOM

Modification des pages visitées — formulaires, contenu, comportement

`scripting.executeScript()`



// LE PROBLÈME

Oui mais… le dossier est sur le disque

- **On manipule Secure Preferences** pour forcer l'installation

  `%LOCALAPPDATA%\Microsoft\Edge\User Data\Default\Secure Preferences`

- **Le dossier de l'extension est copié sur la machine**

  `%LOCALAPPDATA%\...\Extensions\<ID>\`

- **background.js malveillant est lisible sur le disque**

  Visible par tout AV/EDR · Récupérable par analyse forensique

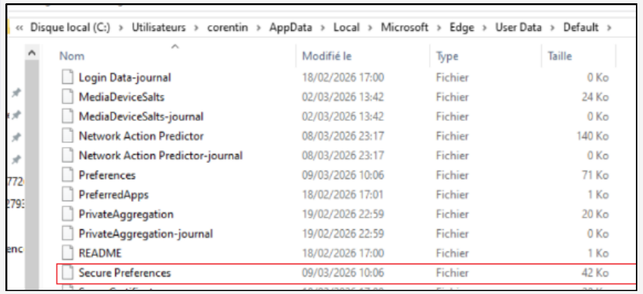
- **En tant qu'attaquant, on veut rester furtif**

  Un artefact malveillant sur le disque, c'est pas terrible

// SUR LE DISQUE

```
Extensions/
  <ID>/
    manifest.json
    background.js   MALVEILLANT
    content.js
```

visible par l'EDR
et l'analyse forensique



// LES PRÉMICES

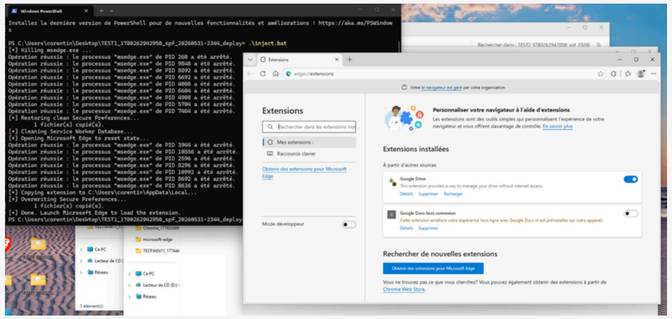
Comportement inattendu lors des tests

Comportement particulier avec 2 extensions ayant le même id et les mêmes noms de fichier…

ACTIVE AGENTS - 2 | Filter agents... | [+] Create Agent | [×] Flush Agents

NAME	ID	IP	BROWSER	LAST CONNECTION	REACHABLE
TEST1	1780262942950	–	edge	Not Yet	false
TEST2	1780262947008	–	edge	Not Yet	false

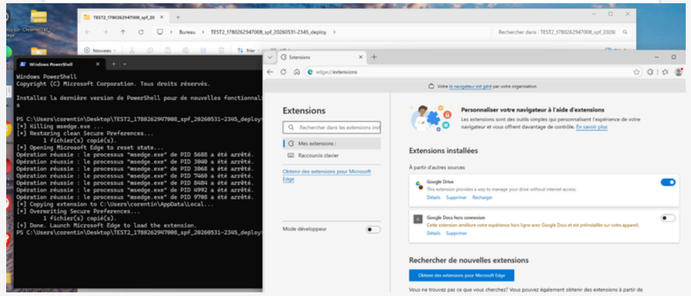
Injection de l'extension TEST1 :



NAME	ID	IP	BROWSER	LAST CONNECTION	REACHABLE
[>] TEST1	1780262942950	192.168.54.2	edge	2026-05-31 23:48:35.436	true
TEST2	1780262947008	–	edge	Not Yet	false

Réception du beacon de l'extension TEST1

Injection de l'extension TEST2 :



NAME	ID	IP	BROWSER	LAST CONNECTION	REACHABLE
[>] TEST1	1780262942950	192.168.54.2	edge	2026-05-31 23:57:08.722	true
TEST2	1780262947008	–	edge	Not Yet	false

Après avoir écrasé la précédente extension, c'est le code de TEST1 qui ping back vers mon C2 alors que j'ai injecté TEST2 ???

// COMMENT ÇA MARCHE EN VRAI

Le Service Worker et son cache

1 **Edge se lance**
Lit Secure Preferences → identifie les extensions installées

2 **Service Worker démarre**
background.js est compilé et stocké en cache
`Service Worker\ScriptCache\`

3 **Enregistrement persistant (LevelDB)**
ID extension → URL script → clé de cache
`Service Worker\Database\`

4 **ScriptCache**
Script compilé indexé par URL — pas par contenu du fichier
`Service Worker\ScriptCache\`

// CLÉ D'INDEXATION DU CACHE

Le ScriptCache est indexé par **URL du script**, pas par son contenu :

```
// schéma
chrome-extension://<ID>/<fichier>
// exemple réel
chrome-extension://nmhdhpibnnopknkmonacoephklnflpho/
  background.js
```

// INSIGHT CLÉ

Au redémarrage, Edge vérifie que le dossier **existe**, puis sert le script depuis le cache — **sans vérifier que le fichier sur disque correspond au cache.**

```
%LOCALAPPDATA%\Edge\...\Service Worker\
  ├ Databases\     ← lien script ↔ Service Worker
  ├ ScriptCache\   ← stockage des scripts compilés
  └ CacheStorage\  ← cache réseau (hors scope)
```

// ET DONC ?

Si on remplace le dossier malveillant par un dossier vide avec le même ID et les mêmes noms de fichiers… le cache n'est jamais invalidé.

manifest.json	background.js	content.js
identique	0B	0B
	VIDE	VIDE

remplace le dossier malveillant · le payload malveillant tourne quand même dans le cache

// GHOST ANCHOR PERSISTENCE — GAP

La chaîne d'infection en 4 étapes

01 **Deploy A**
Injection de l'extension malveillante A via Secure Preferences + HMAC recalculé
`stomp`

02 **Trigger Registration**
Ouvrir Edge → service worker enregistré en DB + background.js compilé en ScriptCache
`obligatoire`

03 **Stomp with B**
Remplacer A par B (même ID, mêmes fichiers, contenu vide) — ScriptCache non invalidé et changement du chemin de déploiement pour cet ID.
`GAP`

04 **Remove A**
Supprimer le dossier de A. Zéro artefact malveillant sur le disque. Payload actif dans le cache.
`fileless ✓`

// VALIDATION

Testé sur Edge, Chrome, Brave, Vivaldi

Browser restart
Service worker restauré depuis ScriptCache à chaque lancement
✓ persiste

System reboot
Registration DB + ScriptCache survivent au redémarrage machine
✓ persiste

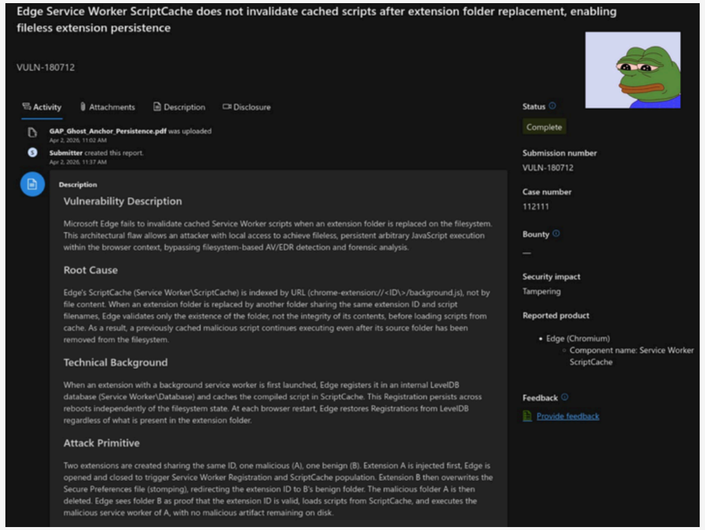
Browser update
Les mises à jour n'invalident pas le cache des extensions locales
✓ persiste

MDE / EDR scan
Zéro alerte levée — dossier propre, payload invisible sur le disque
✓ indétecté

Microsoft à la réception du rapport

// RESPONSIBLE DISCLOSURE · MSRC CASE #112111

Microsoft dit "not a bug". Normal.


Edge Service Worker ScriptCache does not invalidate cached scripts after extension folder replacement, enabling fileless extension persistence

VULN-180712

Activity Attachments Description Disclosure

GAP_Ghost_Anchor_Persistence.pdf was uploaded
Apr 2, 2026, 11:02 AM

Submitter created this report.
Apr 2, 2026, 11:37 AM

Description

Vulnerability Description

Microsoft Edge fails to invalidate cached Service Worker scripts when an extension folder is replaced on the filesystem. This architectural flaw allows an attacker with local access to achieve fileless, persistent arbitrary JavaScript execution within the browser context, bypassing filesystem-based AV/EDR detection and forensic analysis.

Root Cause

Edge's ScriptCache (Service Worker\ScriptCache) is indexed by URL (chrome-extension://<ID\>/background.js), not by file content. When an extension folder is replaced by another folder sharing the same extension ID and script filenames, Edge validates only the existence of the folder, not the integrity of its contents, before loading scripts from cache. As a result, a previously cached malicious script continues executing even after its source folder has been removed from the filesystem.

Technical Background

When an extension with a background service worker is first launched, Edge registers it in an internal LevelDB database (Service Worker\Database) and caches the compiled script in ScriptCache. This Registration persists across reboots independently of the filesystem state. At each browser restart, Edge restores Registrations from LevelDB regardless of what is present in the extension folder.

Attack Primitive

Two extensions are created sharing the same ID, one malicious (A), one benign (B). Extension A is injected first, Edge is opened and closed to trigger Service Worker Registration and ScriptCache population. Extension B then overwrites the Secure Preferences file (stomping), redirecting the extension ID to B's benign folder. The malicious folder A is then deleted. Edge sees folder B as proof that the extension ID is valid, loads scripts from ScriptCache, and executes the malicious service worker of A, with no malicious artifact remaining on disk.

Status
Complete

Submission number
VULN-180712

Case number
112111

Bounty
—

Security impact
Tampering

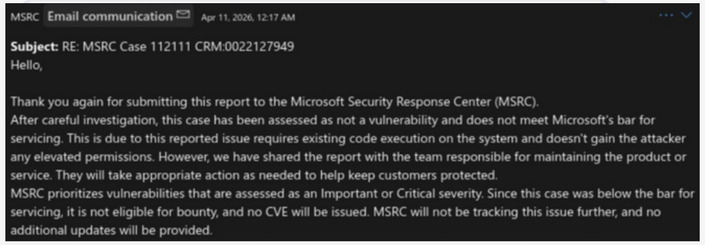
Reported product
- Edge (Chromium)
  - Component name: Service Worker ScriptCache

Feedback
Provide feedback

Rapport soumis au MSRC

// RÉPONSE MSRC — 11 AVRIL 2026

"This case has been assessed as **not a vulnerability** and does not meet Microsoft's bar for servicing. […] it is **not eligible for bounty**, and **no CVE will be issued**."


MSRC Email communication Apr 11, 2026, 12:17 AM

Subject: RE: MSRC Case 112111 CRM:0022127949
Hello,

Thank you again for submitting this report to the Microsoft Security Response Center (MSRC).
After careful investigation, this case has been assessed as not a vulnerability and does not meet Microsoft's bar for servicing. This is due to this reported issue requires existing code execution on the system and doesn't gain the attacker any elevated permissions. However, we have shared the report with the team responsible for maintaining the product or service. They will take appropriate action as needed to help keep customers protected.
MSRC prioritizes vulnerabilities that are assessed as an Important or Critical severity. Since this case was below the bar for servicing, it is not eligible for bounty, and no CVE will be issued. MSRC will not be tracking this issue further, and no additional updates will be provided.

VULN-180712 · Soumis le 2 avril 2026 · Statut : Complete · Bounty : —

Normal.

// FIN DE LA RUMP

Ghost Anchor Persistence
Chromium-based Vulnerability

github.com/ Fir3n0x/GAP

fir3n0x. github.io

x.com/ fir3n0x_

Merci pour votre attention · des questions ?