Bitwarden Network Security Assessment Report

ISSUE SUMMARIES, IMPACT ANALYSIS, AND RESOLUTION

BITWARDEN, INC

Table of Contents

Summary

In May 2022, Bitwarden engaged with cybersecurity firm Cure53 to perform penetration testing and develop a detailed and encompassing security assessment across Bitwarden IPs, servers, and web applications. A total of 10 days were invested to reach total coverage needed for Bitwarden.

The overall impression from Cure53 is that Bitwarden, including the network infrastructure and web applications that power the product, exhibits a strong security foundation with zero exploitable vulnerabilities found. Four issues were discovered with Cure53 designating two being a low security threat and the other two being designated as informational-only.

This report was prepared by the Bitwarden team to cover the scope and impact of the four issues found by the Cure53 team and expected resolution steps. For completeness and transparency, a copy of the report delivered by Cure53 has also been attached to this report.

Issues

BWN-03-001 WP1: Direct access to Azure App Services

It was discovered the API web server exposes its host origin at Azure through a response cookie set under certain requests. By knowing the origin domain of the webserver, protections such as Cloudflare can be bypassed by a potential attacker.

Resolution

Status: Issue has been fixed post-assessment.

IP Restrictions and additional networking rules have been added to underlying app services to ensure requests have been scrutinized before reaching the services.

BWN-03-002 WP1: Direct access to Kubernetes services

Bitwarden operates an “icon service” hosted at icons.bitwarden.net. It was discovered through subdomain scanning that an alternate service exposed the host IP address of the same Kubernetes cluster being used by the icon service, which can lead to an attacker being able to subvert protections provided by services such as Cloudflare.

Resolution

Status: Issue has been fixed post-assessment.

The underlying resources have been properly proxied and migrated off of the old IP address.

BWN-03-003 WP2: 2FA brute-force protection based on rate limits

Brute-force attacks on two-factor authentication codes are prevented by Cloudflare rate limits. Attackers can circumvent this if they have access to an extensive set of proxies via a bot net, for example, that can then help them in identifying the correct 2FA code.

Resolution

Status: Issue has been fixed post-assessment

A captcha challenge was introduced when there is a certain number of failed login attempts, either from master password attempts or from two-factor authentication attempts. In addition, all traffic to authentication endpoints will be routed through Cloudflare, which will also enforce rate limits.

BWN-03-004 WP2: Cross-Origin-related HTTP security headers missing

It was discovered that Bitwarden is not implementing some of the newer security headers such as Cross-Origin Resource Policy (CORP), Cross-Origin Opener Policy (COOP), Cross-Origin Embedder Policy (COEP).

Resolution

Status: Not addressed but actively investigating.

We are investigating the impacts of adding CORP, COEP, COOP security headers and following new developments closely.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Identified Vulnerabilities

The following sections list all vulnerabilities and implementation issues identified throughout the testing period. Please note that findings are listed in chronological order rather than by their degree of severity and impact. The aforementioned severity rank is simply given in brackets following the title heading for each vulnerability. Furthermore, each vulnerability is given a unique identifier (e.g., *BWN-03-001*) to facilitate any future follow-up correspondence.

Miscellaneous Issues

This section covers any and all noteworthy findings that did not lead to an exploit but might assist an attacker in successfully achieving malicious objectives in the future. Most of these results are vulnerable code snippets that did not provide an easy way to be called. Conclusively, while a vulnerability is present, an exploit might not always be possible.

BWN-03-001 WP1: Direct access to Azure App Services *(Low)*

Testing confirmed that the host *vault.bitwarden.com* REST API discloses a domain name concerning an internal Azure App Service. Specifically, the *api.bitwarden.com* domain is mapped to the *bitwardenapi-s38gsxx.azurewebsites.net* App Service and inserts an additional layer of protection using Cloudflare. By directly accessing the *bitwardenapi-s38gsxx.azurewebsites.net* domain, one can bypass the Cloudflare protection.

The domain is disclosed via a cookie, as can be observed in the following response header.

PoC cURL command 1:

```
curl -i https://vault.bitwarden.com/api/now

HTTP/2 200
date: Thu, 05 May 2022 07:18:56 GMT
content-type: application/json; charset=utf-8
cf-ray: 7067b47a0a2dfa7c-AMS
set-cookie: TiPMix=84.2580808253298; path=/; HttpOnly; Domain=bitwardenapi-
s38gsxx.azurewebsites.net; Max-Age=3600; Secure; SameSite=None, x-ms-routing-
name=self;
[...]
```

To verify the direct access to the internal API, one need only execute the following cURL command.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

PoC cURL command 1:

```
curl http://bitwardenapi-s38gsxx.azurewebsites.net/now

"2022-05-05T07:35:54.3353999Z"
```

Even in the eventuality that only one domain name is disclosed in this scenario, Cure53 advises preventing public access to all App Service domains. This can be achieved by configuring an IP allow-list that only contains the intended Cloudflare IPs. Furthermore, the development team could consider either removing the *TiPMix* cookie or setting the domain to the intended value.

BWN-03-002 WP1: Direct access to Kubernetes services *(Low)*

Whilst enumerating subdomains, several domains were found to be proxied through Cloudflare. However, testing confirmed that the *appfunctions.bitwarden.net* domain points at a different IP address, https://20.62.225.137/, which responds with a Kubernetes ingress SSL certificate. Host-header enumeration of known Bitwarden subdomains against the ingress host leads to two service responses, namely:

- *icons.bitwarden.net*
- *func.bitwarden.com*

Example request:
This request is sent to 20.62.225.137 port 80 rather than the IP returned from DNS.

```
GET / HTTP/1.1
Host: icons.bitwarden.net
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/100.0.4896.127 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/
webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Connection: close
```

The response provided demonstrates that the icon service is hosted in this Kubernetes cluster.

```
HTTP/1.1 308 Permanent Redirect
Date: Thu, 05 May 2022 08:47:41 GMT
Content-Type: text/html
Content-Length: 164
Connection: close
Location: https://icons.bitwarden.net
```

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

```
<html>
<head><title>308 Permanent Redirect</title></head>
[...]
```

Typically, Cloudflare aggressively caches the icon requests and implements a basic WAF. However, bypassing this essential Cloudflare process allows any malicious user to directly utilize and attack the icon service.

BWN-03-003 WP2: 2FA brute-force protection based on rate limits *(Info)*

Bitwarden permits the configuration of two-factor authentication based on emails. Here, testing confirmed that the enumeration of all possible email 2FA codes is only prevented by the current Cloudflare rate limits in place. Therefore, attackers with access to an extensive set of proxies via a bot net, for example, may be able to identify the correct code. In addition, in the eventuality that the Cloudflare protection can be bypassed with direct access to the App Service domain - as documented in ticket BWN-03-001 - the aforementioned 2FA codes can be enumerated more efficiently.

In order to enforce rate limits, one can advise ensuring that public access to the Azure App Services is denied to such an extent that the Bitwarden vault is only accessible through Cloudflare, particularly in relation to the authentication endpoint.

BWN-03-004 WP2: Cross-Origin-related HTTP security headers missing *(Info)*

Since Bitwarden constitutes a password manager application, the framework exhibits a higher risk in relation to critical data leakage via modern attacks such as Spectre[1]. Here, the discovery was made that the Bitwarden application lacks several of the newer[2] Cross-Origin-infoleak-related HTTP security headers in their responses. This does not directly lead to a security issue, yet it might aid attackers in their efforts to exploit other areas of weakness, such as those relating to the aforementioned Spectre attack scenario[3]. The following list enumerates the headers that require review in order to prevent associated vulnerabilities.

- **Cross-Origin Resource Policy (CORP)** and *Fetch Metadata Request* headers allow developers to control which sites can embed their resources, such as images or scripts. They prevent data from being delivered to an attacker-controlled browser-renderer process, as seen in *resourcepolicy.fyi* and *web.dev/fetch-metadata*.

[1] https://meltdownattack.com/
[2] https://security.googleblog.com/2020/07/towards-native-security-defenses-for.html
[3] https://meltdownattack.com/

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

- **Cross-Origin Opener Policy (COOP)** grants developers the ability to ensure that their application window will not receive unexpected interactions from other websites, allowing the browser to isolate it in its own process. This adds important process-level protection, particularly in browsers that do not enable full Site Isolation; see *web.dev/coop-coep.*
- **Cross-Origin Embedder Policy (COEP)** ensures that any authenticated resources requested by the application have explicitly opted-in to passing into load state. In the current climate, to guarantee process-level isolation for highly sensitive applications in Chrome or Firefox, applications must enable both COEP and COOP; see *web.dev/coop-coep.*

Generally speaking, the absence of Cross-Origin security headers should be considered a negative practice that could be avoided in times when attacks such as Spectre are known to be well-exploitable and exploit code is publicly available. It is recommended to insert the aforementioned headers into every relevant server response. Resources with detailed information regarding headers of this nature are available online, explaining both header-setup best practices[4] and the potential consequences of bypassing setup entirely[5].

[4] https://scotthelme.co.uk/coop-and-coep/
[5] https://web.dev/coop-coep/