Fine penetration tests for fine websites

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

Pentest-Report ExpressVPN Extension 10.-11.2018

Cure53, Dr.-Ing. M. Heiderich, MSc. N. Kobeissi, M. Kinugawa, Dipl.-Ing. A. Inführ

Index

Introduction

“Easily secure all of your web activity with a Google Chrome VPN add-on”

From https://www.expressvpn.com/vpn-software/chrome-vpn

This report documents the results of a penetration test and source code audit against the ExpressVPN browser extension for Chrome. The assessment was carried out by Cure53 in October 2018 and yielded eight security-relevant findings. A fix verification process was performed in mid-November 2018.

It should be noted that the test target delineated above - that is the ExpressVPN browser extension for Chrome - was analyzed in separation, as the scope did not cover VPN servers, configurations etc. This means that four members of the Cure53 team, who examined the scope over the course of roughly seven days in Calendar Week 42 in 2018, have dedicated all efforts to investigations targeting the extensions.

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

To facilitate efficient analyses and reaching maximum coverage, Cure53 was given access to the sources of the extension, as well as provided with builds created by the ExpressVPN. In addition, the ExpressVPN team shared a presentation that highlighted the most relevant parts of the scope.

With very good preparations and the clear documentation of the scope furnished to Cure53 by the ExpressVPN team, the assessment went very smoothly, with neither technical nor any other obstacles in sight. A shared Slack channel was used for communications between the in-house team at the ExpressVPN compound and the Cure53 testing team. In this channel, Cure53 shared progress reports about the audit. Still, not much communication was needed because the scope was documented in a very clear manner.

Among eight spotted issues, four were categorized as vulnerabilities and four were marked as general weaknesses. In terms of potential implications of the findings, not a single item among the discoveries had received a severity level higher than *“Medium”.* Quite clearly, this is a good security indicator.

A fix verification process was initiated mid-November 2018 and yielded positive results as well. All issues documented in this report have been amended with fix notes for full transparency.

In the following sections, the report first sheds light on the scope, describing all noteworthy findings together with possible mitigation routes. The report then closes with broader conclusions as Cure53 shares the project team’s impressions about the general security and privacy posture of the ExpressVPN browser extension in scope.

Scope

- **ExpressVPN Chrome browser extension**
  - Sources were shared with Cure53
  - Builds were shared with Cure53

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

Identified Vulnerabilities

The following sections list both vulnerabilities and implementation issues spotted during the testing period. Note that findings are listed in a 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. Each vulnerability is additionally given a unique identifier (e.g. *EXP-01-001*) for the purpose of facilitating any future follow-up correspondence.

EXP-01-003 Extension: Manifest does not specify *match_about_blank* (Medium)

The deployed Manifest file does not specify the *match_about_blank*[1] key for content scripts. This key normally ensures that content scripts are injected into frames that have their URLs set to either *about:blank* or *about:srcdoc*. A malicious web page can abuse this behavior to bypass the GPS and fingerprint protections. Relevant Proofs-of-Concept (PoCs) are provided next.

PoC for *About:blank*

```
<iframe src="about:blank">
</iframe>
<script>
console.log(window.frames[0].navigator.geolocation.watchPosition+"") // <--
access to unmodified window
// > function watchPosition() { [native code] }
</script>
```

PoC for *About:srcdoc*:

```
<iframe srcdoc="<script>console.log(navigator.geolocation.watchPosition)</
script>"></iframe>
// > function watchPosition() { [native code] }
```

It is highly recommended to specify *"match_about_blank": true* in the extension's Manifest file for content scripts. This will guarantee that all origins are protected.

Note: *This issue was fixed by the ExpressVPN team and the fix was then verified successfully by Cure53 on 13th of November 2018.*

[1] https://developer.mozilla.org/en-US/docs/Mozilla/Add-ons/WebExtensions/manifest...tch_about_blank

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

EXP-01-004 Extension: Spoofed *Geolocation* object not frozen *(Medium)*

The ExpressVPN extension overwrites the functions of the *Geolocation* object to be able to protect the end-user by returning fake *geolocation* values. It was discovered that the extension is not using *Object.freeze* to secure the *Geolocation* object after the modifications are applied. A malicious website can simply use *delete* to remove the hooks and get access to the native functions again.

JavaScript Proof-of-Concept code:

```
console.log(navigator.geolocation.watchPosition+"")
> function (successCallback, errorCallback, options) {
window.uJIYdqo = successCallback;
window.vzyzVnG = errorCallback;
window.zCMck = options;
waitWatchPosition();
}

delete navigator.geolocation.watchPosition
console.log(navigator.geolocation.watchPosition+"")
> function watchPosition() {
[native code]
}
```

It is recommended to use ES5's *Object.freeze*[2] to prevent the deletion of hooks.

Note: *This issue was fixed by the ExpressVPN team and the fix was then verified successfully by Cure53 on 13th of November 2018.*

EXP-01-005 Extension: Malware URL detection can be bypassed *(Low)*

The ExpressVPN extension is trying to detect either crypto-mining, malware- or Phishing-related domains by checking the loaded domain against a list of known malicious web pages. It was discovered that this check can be bypassed by specifying user-information in front of the malicious domain.

Proof-of-Concept:

```
<script>
location = "http://a:a@amazon.co.uk.security-check.ga/"
</script>
```

File:
Build/dev/chrome/scripts/background.js

[2] https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object/freeze

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

Code:

```
// details.url contains username + password URL part
for (var i = 0; i < blockChecks.length; i++) {
var shouldBlock = ABPFilterParser.matches(blockChecks[i].parsedFilterData,
details.url, { domain: currentPageDomain, elementTypeMaskMap:
ABPFilterParser.elementTypes.SCRIPT });
```

Proposed fixed:

```
function blockBadRequests(details) {
var currentPageDomain = new URL(details.url).hostname;
var tmp_url = new URL(details.url);
details.url = tmp_url.origin + tmp_url.pathname + tmp_url.search
```

It is recommended to evaluate the proposed fixed and apply it to the *blockBadRequests* logic. By parsing the URL and removing all user or password information, it is ensured that the deployed blacklist cannot be easily bypassed.

Note: *This issue was fixed by the ExpressVPN team and the fix was then verified successfully by Cure53 on 13th of November 2018.*

EXP-01-006 Extension: Non-HTML documents bypass *content* scripts *(Medium)*

The deployed *content* scripts ensure that injection possibility only exists for HTML documents. This is implemented by checking the *root* element and verifying that it is *'html'*. In case a malicious web page is loading a non-HTML document like SVG or XHTML, the ExpressVPN's *content* scripts will not apply their protection. The web page is still able to use JavaScript and therefore attack the user's privacy.

XHTML Proof-of-Concept:

```
<?xml version="1.0"?>
<body xmlns='http://www.w3.org/1999/xhtml'>
<script>
console.log(document.documentElement.tagName.toLowerCase()) // > body
</script>
</body>
```

SVG Proof-of-Concept:

```
<svg xmlns="http://www.w3.org/2000/svg">
<script>
console.log(document.documentElement.tagName.toLowerCase()) // > svg
</script>
</svg>
```

Affected File:
Build/dev/chrome/scripts/content/fingerprinting.js

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

Code:

```
function isHtml() {
return (document.documentElement.tagName.toLowerCase() === 'html');  // only
inject in html documents
}
```

File:
Build/dev/chrome/scripts/content/Gps.js

Code:

```
function shouldInject() {
return (document.documentElement.tagName.toLowerCase() === 'html');  // only
inject in html documents
}
```

It should be taken into consideration to adapt the content scripts so that it supports non-HTML documents like SVG or XHTML. This ensures a malicious web page cannot simply bypass the client-side protections by loading these kind of document resources.

Note: *This issue was fixed by the ExpressVPN team and the fix was then verified successfully by Cure53 on 13th of November 2018.*

Miscellaneous Issues

This section covers those noteworthy findings that did not lead to an exploit but might aid an attacker in achieving their malicious goals 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.

EXP-01-001 Extension: Manifest contains unused permissions *(Info)*

It was discovered that the *“proxy”* and *“contextMenus” p*ermissions are specified in the manifest file but are never used by the extension. As these permissions are not required, they should simply be removed.

Note: *This issue was not yet addressed. The reasons were discussed with the Cure53 team and the proposed solution was mutually accepted.*

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

EXP-01-002 Extension: Lists allow third-parties to affect functionality *(Low)*

The ExpressVPN extension loads three external lists that contain rule-sets for blocking cryptocurrency miners and malware. These lists are then parsed into a set of rules, wherein matching URLs are blocked accordingly.

This setup allows three distinct third-parties (AdBlock, AdBlock Plus and Phishtank) unnecessary levels of arbitrary control, specifically held over the browsing capabilities of users who chose to install the ExpressVPN extension.

In order to minimize the level of direct control given to these third-parties and pertinent to the users, we recommend that the lists should first be locally downloaded into servers controlled fully by the ExpressVPN staff. This would mean that an arbitrary change in the rule-sets would first have to be merged into the ExpressVPN's copies before possibly affecting users. ExpressVPN could also choose to implement vetting or review processes into the contents of these updates before merging them into their local copies of the rule-sets.

Affected File:
source/scripts/background.js

Affected Code:

```
function downloadFilters(cb) {
	let promises = [];
	promises.push(downloadAndParseFilter('https://raw.githubusercontent.com/
hoshsadiq/adblock-nocoin-list/master/nocoin.txt', '/filters/nocoin.txt',
parsedFiltersData.parsedCoinFilterData));
	promises.push(downloadAndParseFilter('https://easylist-
downloads.adblockplus.org/malwaredomains_full.txt', '/filters/malware.txt',
parsedFiltersData.parsedMalwareFilterData));
	promises.push(downloadAndParseFilter('https://data.phishtank.com/data/
online-valid.csv', '/filters/phishing.txt',
parsedFiltersData.parsedPhishingFilterData));
```

Note: *This issue was mitigated by the ExpressVPN team and the mitigation was then verified successfully by Cure53 on 13th of November 2018.*

EXP-01-007 Extension: *Content* scripts bypassed through *Blob* document *(Info)*

It was found that the WebExtension's *content* scripts in Chrome are not injected into Blob documents and this behavior can be used for bypassing the ExpressVPN extension's protection. The following PoC shows that "*Spoof your location*" protection can be bypassed via the Blob document.

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

PoC:

```
<script>
var blob = new
Blob(['<script>alert(navigator.geolocation.watchPosition+"")//function
watchPosition() { [native code] }<\/script>'],{type : 'text/html'});
location = window.URL.createObjectURL(blob);
</script>
```

Until browser vendors fix this behavior, there is currently no easy solution to this problem. Since this behavior has been reported to the Chrome team long ago and no fix was deployed so far, it is expected that the behavior in question is here to stay.

Note: *This issue was fixed by the ExpressVPN team and the fix was then verified successfully by Cure53 on 13th of November 2018.*

EXP-01-008 Extension: No *data:URI* support for *content* scripts *(Info)*

Major browsers currently do not support loading of *content* scripts into documents created via the *data:* protocol handler[3]. Although browsers treat these kinds of documents as having an insecure (and therefore anonymous) origin, they can still harm the security of a web extension. In case of ExpressVPN, this behavior can be abused to bypass the fingerprinting detection.

Proof-of-Concept:

```
<iframe
src="data:text/html,<script>alert(document.createElement('canvas').toBlob)</
script>"></iframe>
```

There is currently no easy solution to this problem. A fix from the browser vendors must be awaited.

Note: *This issue was mitigated by the ExpressVPN team and the mitigation was then verified successfully by Cure53 on 13th of November 2018.*

[3] https://developer.mozilla.org/en-US/docs/Mozilla/Add-ons/WebExtensions/match_patterns#scheme_2

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

Conclusions

The results of this Cure53 assessment of the ExpressVPN browser extension for Chrome are positive, and the mid-November 2018 fix verification process confirms that. Yet there is a small caveat to consider. On the one hand, four Cure53 testers who took part in this project and investigated the scope over the course of seven days in October 2018, can conclude that the narrowly scope extension item seems safe in a general sense. On the other hand, before the first fixes came in, there were still many areas with unresolved, minor security details. The latter was also connected with the fact that the ExpressVPN tasks itself with very ambitious goals in terms of what it wants to protect against.

To reiterate, the ExpressVPN browser extension is well-implemented and it is apparent that the usual VPN bypass tricks, as well as actively malicious web pages trying to unmask the user, were taken into consideration and mitigated. The communication between *content* scripts and the background page (and therefore the native application) made a good impression as well. It needs to be underlined that no security issues which would allow an attacker to influence the state of the VPN connection via a malicious web page or alike were discovered. Note however that several features that initially aimed to offer better privacy for users but fell victim to browser-based shortcomings were removed by the ExpressVPN team after this test. While this seems radical at first, this is considered a positive development, since it is better not to promise privacy features in the first place when the browser bypasses the efforts of the extension developers anyway. Moving on to specifics, not all features provided in the WebExtension's Manifest were used for *content* scripts, resulting in a simple bypass described in EXP-01-003. Additionally, some minor mistakes - like having forgotten to freeze the geolocation object - would have let a malicious web page subvert the deployed privacy protection. Lastly, some bugs regarding the general *content* script implementation in web browsers affected the ExpressVPN users and hindered the browser extension from properly deploying custom client-side privacy preserving logic.

All in all, the impression gained about the ExpressVPN WebExtension is positive and the project complies with the major security and privacy standards. Most of the described issues, which largely constitute minor mistakes in the WebExtension's code, were easily fixed. Nevertheless, modern browsers still suffer from issues in the underlying layers, that is in the WebExtension's design itself. These can only be fixed by the browser vendors. Otherwise, a WebExtension cannot properly implement client-side protections via *content* scripts.

Cure53 would like to thank the ExpressVPN team for their excellent project coordination, support and assistance, both before and during this assignment.