Fine penetration tests for fine websites

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

TunnelBear Security Assessment Summary 10.2018

Cure53, Dr.-Ing. Mario Heiderich & Team

Introduction

This technical summary report describes the results of a series of major VPN & application and server security audits performed by the Cure53 team in June 2018 against the TunnelBear project.

The security assessment described here, as well as this particular document, were requested by the maintainers of the TunnelBear compound. It needs to be noted that this is a second summary of this type provided by Cure53, as the first one was issued back in the summer of 2017 and followed-up on the outcomes of the assessments carried out since 2016. Due to continuous security-centered cooperation, signifying audits and tests against various items in the TunnelBear scope performed in 2018, Cure53 is happy to issue an updated account and verdict on the current security posture of the TunnelBear project.

One important change to point out in the overall setup is that TunnelBear, which generally seeks to offer privacy to a broad range of users who rely on its access to VPN servers, has been recently acquired by McAfee, LLC. Nevertheless, the maintainers of the TunnelBear project still engage in regular, externally-commissioned security audits.

Test Summary & Methodology

The findings reported here pertain to the overall fifth security project completed by Cure53 for TunnelBear. Putting the project in context, this longer-term cooperation becomes increasingly advantageous in that the Cure53 testers involved in the investigations have acquired tremendous knowledge and familiarity with the important security aspects of the TunnelBear compound. Moreover, the methodology has been well battle-tested and, as a consequence, there is consistency in the approaches employed for testing in a logical manner.

The large-scale assessment at hand entailed a penetration test and a security audit of the pre-selected, various entities belonging to the TunnelBear software and service compound. In June 2018, eight members of the Cure53 spent thirtyfour days on the test-targets and made twenty-two security-relevant discoveries. The team was comprised on the basis of the optimum skills possessed by its members and their proven capacity of addressing the assessment’s goals.

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

In terms of the objectives, the main goal was to gain a comprehensive portrait of the scope with an in-depth, extensive security review. The TunnelBear team expressed interest in knowing how well the security promises they have made actually hold against determined attackers. Further, TunnelBear wished to find out whether a determined attacker could compromise their entire network.

Due to the large-scale nature of the project and in order to facilitate progress updates with reference to coverage, the scope was split into several work packages (WPs). These specifically set out to cover Client Applications (WP1), Browser Extensions (WP2), VPN Infrastructure (WP3), Filter Pods (WP4), backend (WP5), and, last but not least, frontend and public websites/servers (WP6). While the test coverage and goals extended to WPs 1-3 and 5-6, the WP4 was ultimately not completed. The reason behind the eventual removal of this item was its lack of test-readiness during the time-window agreed as the period for the Cure53's testing. Otherwise, the assessment adhered to the devised schedule.

As already noted, the Cure53 team utilized the same methodology as before, namely in relying on the so-called white-box approach. This method signifies that Cure53 had access to running applications, binaries, sources and extensive documentation. The Cure53 team was further given SSH access to a *jumphost* so as to be able to investigate the server setup, infrastructure configurations, as well as assess the more general security posture. A clear benefit of the chosen premise concerned open communications between the Cure53 testers and the in-house team at TunnelBear. With the use of a dedicated Slack channel, all arising questions could be answered promptly, precisely and effectively, often through live exchanges. In addition, Cure53 executed live-reporting and filed all findings into the TunnelBear's *JIRA* bug tracker.

Audit Results

From a security standpoint, this assessment revealed great progress that the TunnelBear has made over the recent years. This is clearly enabled by the incessant efforts of the in-house team, which manages to introduce subsequent improvements to the overall integrity and security, despite the high-complexity and size of the compound. Having said that, Cure53 still managed to find twenty-two security-relevant issues.

Checking the number of problems in different categories reveals that twelve issues were classified as vulnerabilities and twelve were deemed general weaknesses. In the table below, all identified issues are presented. A primary focus is place on the notable issues with "*Critical"* and "*High"* severities.

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

Vulnerability	Description
Critical (2)	
OSX Client	The problem allowed the user to escalate to *root*. In particular, the *tbeard root daemon* implemented XPC methods to execute and kill processes. The XPC could be used directly by a malicious program to execute other programs because it verified the signature of the calling process to be that of the TunnelBear agent. However, the agent used this XPC mechanism to launch other programs such as *openvpn* from the app bundle. The flaw could be abused by simply replacing the *openvpn* binary in the bundle with any malicious payload. When TunnelBear seeks to establish a VPN connection and sends the XPC to the *tbeard daemon*, the result is an execution of the malicious program. The issue relied on the fact that the files are owned by the *current* user and could, therefore, be modified.
Windows Client	An equivalent to the aforementioned OSX issue on the Windows environment has also been found by Cure53. While the architecture of the services is slightly different, Windows also implements a privileged maintenance service. In this context, an attacker could replace or modify the *.bat* scripts to become an *administrator*. If TunnelBear was installed in the default *install* directory, then this attack would have been mitigated by the security settings of the "*Program Files*"-folder, which prevent regular users from modifying files. However, when TunnelBear was not installed under this protected path, the user could have modified the scripts and acquire privileges at the *administrator*-level.
High (5)	
OSX Client	On Mac environments, the TunnelBear is running a TunnelBear agent called *tbeara.* This agent is listening for XPC calls. One function exposed in this realm is *restartApplicationFromNew-Location*, which basically just executes a program at the path given via the parameter. The agent in question does not run as *root*, so it appeared to be a non-issue at first. However, other users on the system, for example a *guest*, could still issue XPC calls and escalate to the user running *tbeara*. In combination with the aforementioned OSX issue, this could be leveraged for upgrading from a *guest* user to *root*.

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

Extension	It was found that the host-matching used in the *PAC* script was not strict and the unintended hosts could be accessed without VPN. This behavior could have been abused for disclosing the real IP address of a user through an MitM attack, provided that the user relied on an untrusted connection.
Extension	The *api.tunnelbear.com* domain is whitelisted to intentionally bypass the VPN access. Therefore, to protect safety of the users, all connections to the *api.tunnelbear.com* must be encrypted. The *api.tunnelbear.com* domain has the *preload* directive in the HSTS header set. It was found that the HSTS *preload* was actually not used since the domain was not listed in the Chromium source code's *preload* list[1]. This means there was no automatic upgrade to HTTPS if the first access before receiving the HSTS header was done with an insecure HTTP connection.
Extension & Website	As mentioned before, the *api.tunnelbear.com* domain is whitelisted to intentionally bypass the VPN access. It was found that FTP requests to *api.tunnelbear.com* are also allowed under this setting. An attacker could obtain the actual IP address of a user when they navigated to the supplied URL. This succeeds because the FTP requests are made in plain-text.
Website	Another discovery could be used to force a victim into logging-in with credentials that are known to the attacker. When a new account is created, the user is automatically logged in with the provided credentials, even though the verification link is yet to be clicked on at this point. Since the *registration* handler did not perform a CSRF check, an attacker could force a victim into creating a new account with known credentials. In other words, all actions performed by the victim after the attack would have been happening in the context of an account the attacker has access to.
Medium (3)	
iOS Client	The iOS client was found leaking the VPN exit-country of the user intending to employ the VPN for reaching clear-text DNS. In addition to this, DNS spoofing of **.lazerpenguin.com* with the use of any fake IP - such as *127.0.0.1* - would result in the TunnelBear client being

[1] https://cs.chromium.org/chromium/src/net/http/transport_security_state_static.json

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

	unable to establish a VPN connection.
Shop / OOS	While passively investigating areas that are slightly out of scope of the engagement, an issue was discovered, which can be used by an attacker to manipulate a user's shopping cart by luring the user onto a malicious website.
Windows Client	The privileged *Maintenance Service* was writing logs in the TunnelBear installation directory. A regular user could alter the files if TunnelBear had not been installed in a protected directory. For example, an attacker could replace the targeted logfiles with symlinks pointing to files that are only modifiable with *administrator* privileges, like the files in *C:\Windows*\.
Low (7)	
Website	The issue here resided within subdomain takeovers comprising a common scenario for active exploitation. This especially holds when external services are configured inside the domain's *CNAME*. For example, the TunnelBear compound uses *agents.tunnelbear.com* with a *CNAME* entry of *tunnelbear.desk.com*. If, at some point, the subscription to *Desk* expires and no action to prevent this takes place, this would leave the domain open to takeovers. In other words, it would allow attackers to host arbitrary content on a TunnelBear subdomain.
Website	It was found possible to check whether a given email address is connected to a TunnelBear account by evaluating the response time during the *login* procedure. The approach was successful because a failed *login* attempt with a valid email address took significantly longer than a failed attempt with a non-existing email address.
Android Client	The audit unveiled that clear-text HTTP links are present in the source code provided for review. This could result in unintended leakage or a compromise of a developer's workstation in situations when the attacker is able to Man-In-The-Middle clear-text HTTP communications.
Website	A very large cookie could be set via the client-side code of the *blog* page. This behavior would lead to client-side DoS since the server then refuses to return regular contents and returns a 400 error instead if the HTTP request's size is overly large.
Website	During the analysis of the new privacy features where customers can

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

	request download of their data, it was noticed that the *Scala* endpoints permitted insecure authentication mechanisms. Although in the normal case cookie authentication is used, it was still possible to send credentials via *GET*.
iOS Client	Another issue could potentially allow access to data stored by the application in the iOS keychain on jailbroken devices. Since the *kSecAttrAccessibleAfterFirstUnlock* protection class was used, the stored keychain items are not protected by the iOS encryption mechanism as long as the user has booted their device and entered their passcode for the first time. This allowed for easier access to the stored data by malicious users or malware.
iOS Client	Another flaw simplifies access to data stored by the application on jailbroken devices because a static key was used for encryption. This allows for easier access to the stored data by malicious users or malware.
Informational (5)	
Website	It was found that the TunnelBear web interface could be disabled by setting a crafted *TB_SESSION* cookie. This behavior could be abused to disable the application if an attacker can inject the arbitrary cookie by using a (subdomain-)XSS vulnerability or comparable attack.
Android Client	It was found that the Android TunnelBear client, specifically *v150,* implements pinning for TLS communications on several domains. On the contrary, it still retrieves the initial configuration from *s3.amazonaws.com* without pinning. This would allow state actors and other attackers with substantial resources to sign arbitrary TLS certificates and modify the configuration file. Thus, this sequence would result an attacker-controlled API server instead of the URL of the legitimate TunnelBear API server. The mitigating factors were nevertheless noticed in that credentials are not sent to this attacker-server as the application constantly crashes when the attack is performed.
Server	During the investigation of the *jumphost* configuration it was found that the installed kernel and several application packages are not up-to-date. In this particular case however, none of the updates were deemed to be security-critical.

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

Server	An issue was discovered that allowed identifying the used server product and version. This enables an attacker to look up known vulnerabilities for the software or generally use this information to deduct which operating system is used. The flaw could be used for deploying or planning further attacks.
Windows Client	Another discovery allowed identifying the user's email address by reading the *trace.log*-file. In a shared working environment, a user can take advantage of this information to identify TunnelBear users.

The deployed fixes held up to verification and scrutiny, while some additional repairs and mitigation strategies were still being developed. It is important to note that Cure53 assisted the TunnelBear team with developing some of the fixes, especially when more fine-tuned recommendations were needed because the choice of a solution was not immediately apparent. Nevertheless, it must be underscored that the Cure53 did not perform a full and complete retest, but rather engaged in a thorough and comprehensive fix verification. In other words, the team investigated the provided *commits* and *diffs* provided by the TunnelBear team. With a future outlook, Cure53 and TunnelBear crucially established the value of embedding security resources and regular audits into the broader projects to prevent recurrence and regressions.

Conclusions

All in all, the security at TunnelBear has once again improved by leaps and bounds. The impression gained by Cure53 in 2018 can by no means be compared to the initial, rather suboptimal results yielded two years prior. In other words, the security baselines are now set at a much higher-level and numerous vital milestones have been clearly achieved by the TunnelBear compound.

This is no small feat because Cure53 factually targeted a vast and nearly all-encompassing scope of the TunnelBear web applications, clients, extensions and the connected core services. While the total of twenty-two issues can be seen by some as an overly excessive number, it must be seen in the context of the sheer size, ambitious threat model and goals that the TunnelBear sets for its products and services. In that sense, the results correspond to what could be expected in terms of outcomes. Cure53 has no doubts about the current accomplishments as fewer issues uncovered in this assessment received "*Critical*" and "*High*" risk-markers. In turn, the project yielded matters concerning defense-in-depth approaches, indicating that it would be much harder for attackers to trivially exploit the compound. A significant majority of the issues could be seen as relatively easy to mitigate and not actually posing great security threats but rather indicating areas for improvement.

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

To comment on some details and the aforementioned room for growth, the TunnelBear application now verges on being impossible to break, with only minor and hard-to-leverage flaws. What remained was mostly small-scale leakages and concerns about whether a potential user-victim could be gullible and/or careless enough to allow a more serious attack. More problems still persisted in the client-side applications, especially for Windows and Mac installations. It must here be absolutely guaranteed that nobody can become a super-user illegitimately. Importantly, the TunnelBear team has successfully deployed fixes, even on the more challenging Mac client. Still, this area should closely monitored, as new issues with OpenVPN permissions might resurface. Further, browser extensions were shown to be prone to de-anonymization through modification of traffic. While this issue was quite concerning, ultimately it has been resolved. Cure53 must also commend the remaining applications, especially Android and iOS, as these presented themselves as very robust.

Finally, Cure53 is extremely satisfied to see that nearly all issues reported in the past installments of security testing projects have been addressed properly. From that follows that the TunnelBear team is not only eager to learn from earlier mistakes or oversight, but also has full capacity to develop and deploy adequate mitigations and fixes.

An optimistic verdict should be that the TunnelBear is quite secure. Still, the maintainers must proceed with caution as risks are always emergent and some aspects of the scope are not yet at the same level of robustness and impenetrability as desired. Because the degrees of hardening and security-awareness on the distinct scope items vary, the TunnelBear team must continue to safeguard their users on various planes. While the issues might be of lower-severity in general, when compared to the outcomes of the earlier testing rounds, it is still paramount for security to become an even more explicit and holistically integrated development and deployment priority.

In sum, the targeted TunnelBear scope should be seen as exhibiting an increasingly improving security standing. The positive sign is that the in-house team shows praiseworthy dedication to eradicating problems and preventing regressions, especially as vulnerabilities are actually discussed and fixed in a best-possible and high-quality manner. The TunnelBear always seeks to find the correct course of action to determine a solution, even if it might not comprise a final step in the process. All these indicators make it apparent that the TunnelBear team wishes to make sure that their customers can take advantage of an actually secure VPN solution. Further security work should concentrate on verifying whether the identified positive trends actually continue and are replicated in future tests.