windscribe

Client Application Code Review

FINAL REPORT

Prepared for: Yegor Sak
Director

Windscribe Limited
9251 Yonge St #8901
Richmond Hill, ON, L4C 9T3
Canada

September 24, 2021

Version:	Final
Prepared for:	Windscribe Limited
Date:	September 24, 2021

Table of Contents

Executive Summary

Windscribe Limited engaged Leviathan Security to perform a time-bound security assessment of its VPN client application, which it provides to customers to enable them to connect to Windscribe-provided VPN servers using a variety of protocols. We performed this assessment from September 13, 2021 until September 24, 2021. Our assessment was performed against the code in GitHub with commit hash **`a35abb033ef16e6f80acfc1796fa91c9559a812d`**, last edited on August 21.

Our objective was to review the source code ahead of a potential open-source release, with two major questions in mind. First, are there software licensing issues that are not addressed by the software as it currently exists? Second, are there deviations from security best practices that are present in the code? For this assessment, we were provided with the source code via the potential public-release repository, as well as contacts with Windscribe to answer any questions during the testing period. Our engagement specifically precluded any dynamic testing, relying exclusively upon manual source code review.

Our review uncovered 2 high, 8 medium, and 3 low-severity findings.

Observations

With regard to licensing issues, Windscribe's application does not currently appear to comply with any of the requirements of any of the software licenses for the open-source software included in its executable, which is of concern. However, the remediation is likely to be very straightforward for each of these issues, by combining an in-application (or other similar functionality) viewer for the copyright and license statements of each included library with the forthcoming open-source release. Accordingly, while Windscribe was noncompliant in the past, it seems likely that a full remediation would limit its exposure in this area.

We also found nine issues relating to security, including two high-severity issues: a macOS-specific local privilege escalation / arbitrary code execution issue, and a set of out of date dependencies for the Windscribe application that include known severe vulnerabilities in its supporting libraries. More broadly, while the codebase is very readable, its history as an (until recently) sole-developer project has led to some limiting choices, including a completely custom build system dependent upon Python 2 (now nearly two years past its end of life date), and a near total lack of comments throughout the codebase. These issues will require significant time to remediate, but are likely to inhibit Windscribe's hoped-for community involvement in future development until they have been solved.

Recommendations

In addition to the remediation efforts discussed in each finding, we recommend that Windscribe engage outside assistance in testing the security of its entire end-to-end platform. This engagement would include dynamic testing of the client application as well as its servers, helping to ensure that the servers cannot attack Windscribe's users, that the application performs as expected in a wider range of situations, and that Windscribe's customer-facing commitments to security and privacy are proven out in the real

world. This engagement would also allow both static and dynamic testing of the forthcoming Linux client application, which Windscribe told us was likely to be released for users very soon.

Vulnerability Classification

Impact

When we find a vulnerability, we assign it one of five categories of severity, describing the potential impact if an attacker were to exploit it:

Informational – Does not present a current threat but could pose one in the future if certain changes are made. To protect against future vulnerabilities, fixing the condition is advisable.

Low – May allow an attacker to gain information that could be combined with other vulnerabilities to carry out further attacks. May allow an attacker to bypass auditing or minimally disrupt availability, resulting in minor damage to reputation or financial loss.

Medium – May allow an attacker inappropriate access to business assets such as systems or servers. There may be impact to the confidentiality or integrity of data, or limited disruption of availability, resulting in moderate damage to reputation or financial loss.

High – May allow an attacker inappropriate access to business assets such as systems or servers. There may be substantial or widespread impact to the confidentiality or integrity of particularly sensitive data, or disruption of availability, resulting in significant damage to reputation or financial loss.

Critical – May allow an attacker to gain persistence, or imminently disrupt functionality or disclose data, resulting in severe reputational damage or financial loss.

Skill Level to Exploit

When we find a vulnerability, we assess how skilled an attacker must be to exploit it:

Simple – Requires minimal understanding of the underlying technology. Tools/ techniques for exploiting the vulnerability can be easily found on the internet.

Moderate – Requires significant expertise, possibly in proprietary information, or access to tools that are not readily available to individuals. The unwitting cooperation of a victim or target may also be required.

Advanced – Requires insider access or access to tools that are not publicly available. Successful exploitation of another vulnerability may be required. Direct interaction with the victim or target may also be required.

Impact Rating (Weight)	Skill Level to Exploit Rating (Weight)		
	Advanced (1)	Moderate (2)	Simple (3)
Critical (4)	4	8	12
High (3)	3	6	9
Medium (2)	2	4	6
Low (1)	1	2	3

Severity	
Critical	10-12
High	7-9
Medium	4-6
Low	1-3

Vulnerability Index

This section represents a quick view into the vulnerabilities discovered in this assessment.

ID	SEVERITY	TITLE	COMPONENT
101212	High	Pinned Out of Date Dependencies	Security Best Practices
101391	High	Local Code Execution as Superuser on macOS	Security Best Practices
101236	Medium	License Noncompliance: LZO	License Review
101238	Medium	License Noncompliance: QT	License Review
101239	Medium	License Noncompliance: stunnel	License Review
101240	Medium	License Noncompliance: c-ares	License Review
101241	Medium	License Noncompliance: OpenVPN, TAP-Win32/TAP-Win64, Windows DDK, NSIS, OpenSSL, SSLeay	License Review
101284	Medium	License Noncompliance: Wireguard	License Review
101291	Medium	Python 2 Has Reached End-of-Life	Security Best Practices
101364	Medium	Non-Encryption Used As Encryption	Security Best Practices
101362	Low	Use of hard-coded credentials	Security Best Practices
101379	Low	Named Pipes Are Not Secret	Security Best Practices
101380	Low	Local Denial of Service	Security Best Practices
101213	Info	Custom Dependency Build System	Security Best Practices
101355	Info	Lack of Comments Throughout Code	Security Best Practices

Observations & Analysis

For the purposes of evaluation, we separated this project into several components based on design documentation and discussions with the service team.

Project Components & Descriptions

We have broken the project down into 2 high-level components below.

SECURITY BEST PRACTICES	Windscribe plans to open source its VPN client application in the near term, and requested that as part of its preparation for doing so, Leviathan evaluate the source code and determine if there were any "vulnerabilities, weaknesses, design flaws, [or] bad patterns" present in the code base. Notably, this effort explicitly excludes any kind of dynamic testing or analysis, as well as any testing where the testers could control a server endpoint, and instead relied exclusively upon manual code review.
LICENSE REVIEW	Windscribe had suffered negative consequences as the result of an open-source license noncompliance issue before the beginning of this engagement; accordingly, they asked Leviathan to evaluate other software used by the Windscribe application for potential noncompliance issues. While Leviathan Security Group is not a law firm and cannot provide legal advice, we are able to evaluate what software is in use and whether Windscribe's application seems generally to comply with the license terms. We recommend that Windscribe consult with its counsel with regard to all legal questions.

Security Best Practices

Windscribe plans to open source its VPN client application in the near term, and requested that as part of its preparation for doing so, Leviathan evaluate the source code and determine if there were any "vulnerabilities, weaknesses, design flaws, [or] bad patterns" present in the code base. Notably, this effort explicitly excludes any kind of dynamic testing or analysis, as well as any testing where the testers could control a server endpoint, and instead relied exclusively upon manual code review.

Threat Analysis

Security flaws in a piece of open-source code can be discovered more easily than in closed-source code, owing simply to the complexity of using standard reverse-engineering tools; accordingly, vulnerabilities here which may be used against Windscribe's userbase should be remediated before the code is released to the world.

Observations

Despite not performing any dynamic testing, we found several concerning issues, including a low-value "encryption" algorithm in place on Windows, a significant number of concerning out-of-date dependencies (including a dependency on Python 2 for the build process), and a High-severity vulnerability on macOS. Following the remediation efforts, we recommend that Windscribe ask Leviathan's Technical Services team for a more thorough test of both its client and server-side systems, including a reproducible and manipulable Staging-class server environment that can be used to attack application users, to help to ensure its business model is secure.

Vulnerabilities

PINNED OUT OF DATE DEPENDENCIES

ID	101212
Component	Security Best Practices
Severity	**High**
Impact / Skill Level	High/Simple
Reference	https://curl.se/docs/security.html https://www.openssl.org/news/vulnerabilities-1.1.1.html
Location	tools/deps/vars/*.yml tools/bin/get-pip.py

Observation

Using software beyond its end of life exposes users to exploitation of any security vulnerabilities discovered in the product after patches are no longer released.
A significant number of dependencies brought in by the build system are not current (versions are specified in the **`tools/deps/vars/*.yml`** files). Versions of dependencies are as follows:

```
Boost: 1.69.0 Current: 1.77.0
C-Ares: 1.17.1 Current: 1.17.2
Curl: 7.67.0 Current: 7.78.0
GTest: 1.10.0 Current: 1.11.0
Jom: 1_1_2 Current: 1_1_3
LZO: 2.10 Current: 2.10 (Up to Date)
OpenSSL: 1.1.1d Current: 1.1.1l
OpenVPN: 2.5.0 Current: 2.5.3
Protobuf: 3.17.3 Current: 3.17.3 (Up to Date)
QT: 5.12.11 Current: 5.12.11 (Up to Date, Not Newest Series)
Stunnel: 5.60 Current: 5.60 (Up to Date)
Wireguard: 0.0.20201118 Current: 0.0.20210424
Zlib: 1.2.11 Current: 1.2.11 (Up to Date)

tools/bin/get-pip.py: 20.3.4 Current: 21.2.3
```

In the cases of Curl and OpenSSL, multiple serious vulnerabilities are known to exist in the versions that are in use by Windscribe.

Impact Rationale:
Ignoring out-of-date software for which security issues are known and which are patched in subsequent versions exposes users to unnecessary risk.

Difficulty Rationale:
Since past security vulnerabilities are made widely known in open-source software to encourage upgrades, an attacker need not use bespoke or unknown exploits when attacking out-of-date software.

PINNED OUT OF DATE DEPENDENCIES

Recommendation

In conjunction with Finding #**101213**, stop manually importing dependencies and instead use system packages. Alternately, commit to a monthly or quarterly update cadence for all dependencies, and update all current dependencies as soon as practicable.

LOCAL CODE EXECUTION AS SUPERUSER ON MACOS

ID	101391
Component	Security Best Practices
Severity	**High**
Impact / Skill Level	High/Simple
Reference	https://cwe.mitre.org/data/definitions/287.html
Location	backend/mac/helper/src/server.cpp:46 backend/mac/helper/src/ipc/helper_security.mm:13 backend/engine/engine/helper/helper_mac.cpp:1171 backend/mac/helper/src/execute_cmd.cpp:5

Observation

When a process running with superuser privileges takes input from a process running without such privileges, it is critical to check the authenticity and validity of such input. Failure to do so allows an attacker with access to userspace to escalate their permissions to the superuser level. An authenticity check cannot consist solely of a non-cryptographically-proven self-attestation.

In addition, multiple processes can write to a file (or file-like object) simultaneously on macOS. **`server.cpp:readAndHandleCommand()`** is the macOS helper app function to take commands from the userspace application (via a named file socket) and process them in the superuser helper application. It calls **`helper_security.mm:verifyProcessId()`** to verify the command before executing it by checking the path and code signing information of a process ID (pid). However, **`verifyProcessId()`** takes the pid from the input string written to the socket. An attacker with local non-privileged access can simply identify the (valid) Windscribe process ID and write it into the socket along with the malicious command; the **`verifyProcessId()`** function has no way to validate that the process which wrote the command and pid is the process whose pid it was. Since the helper app will execute commands directly by calling them on the command line with now-attacker-controlled strings (see **`backend/mac/helper/src/execute_cmd.cpp:5-17`**, this allows arbitrary local code execution as the superuser, including local privilege escalation.

Impact Rationale:
This finding can be used to execute arbitrary commands with superuser privileges.

Difficulty Rationale:
This finding can be exploited using **`cat`** or similar basic tools.

Recommendation

Do not rely on self-attestation of the writing process, and implement a different validation method. One possibility is to use a macOS Keychain restricted to the particular application that contains a private key certificate that can be used (through public key cryptography) to validate the identity of the calling application without exposing the private key to other applications; however, this may introduce additional complications, and there may be other appropriate solutions.

PYTHON 2 HAS REACHED END-OF-LIFE

ID	101291
Component	Security Best Practices
Severity	**Medium**
Impact / Skill Level	Medium/Simple
Reference	https://www.python.org/doc/sunset-python-2/
Location	tools/deps/*

Observation

Using software beyond its end of life exposes users to exploitation of any security vulnerabilities discovered in the product after patches are no longer released.
The custom build system (see Finding #**101213**) relies upon Python 2, which reached its end of life nearly two years ago (January 1, 2020). According to the Python team, "That means that we will not improve it anymore after that day, even if someone finds a security problem in it."

Impact Rationale:
Security patches are no longer being released for Python 2; hence, developers required to persist Python 2 on their machines are exposing themselves to unnecessary risk.

Difficulty Rationale:
Since patches are no longer being released for Python 2, a vulnerability need not be new or bespoke to exploit it.

Recommendation

Upgrade all scripts to Python 3 compatibility and update related documentation.

NON-ENCRYPTION USED AS ENCRYPTION

ID	101364
Component	Security Best Practices
Severity	**Medium**
Impact / Skill Level	Medium/Simple
Reference	https://cwe.mitre.org/data/definitions/311.html
Location	backend/engine/helper/simple_xor_crypt.cpp backend/windows/windscribe_service/simple_xor_crypt.cpp

Observation

Unencrypted data in transit is subject to both information disclosure and data tampering. Unencrypted data at rest is vulnerable to disclosure if an attacker can obtain read access to the file.

Well-known mathematical techniques allow an attacker who can intercept multiple messages XORed with the same key to recover the key. This is the technique that enables true one-time pads to be secure, but in the case of their reuse, renders them useless. It works as follows:

Take two messages, m and n. They are transformed using the bitwise XOR operation (denoted ^) into ciphertexts C and D, where C == m ^ KEY and D == n ^ KEY. They are then sent over the transmission medium.

When an attacker recovers C and D, the attacker can XOR them together. C ^ D == m ^ KEY ^ n ^ KEY == m ^ n -- that is, the result is the two plaintexts, XORed. When the plaintexts are identifiable (e.g., through being standard English words, as they are in the case of the VPN client communications), separating m and n from m^n is trivial (using, e.g., a potential or partial wordlist to identify likely candidates), and produces both plaintexts. KEY is then recoverable because C ^ m == KEY. The key can then be used to retrieve any future plaintext, or to encode any future message.
simple_xor_crypt.cpp provides an **encrypt** function which does not encrypt anything. Instead, it uses a bitwise XOR function, as follows:

```
std::string SimpleXorCrypt::encrypt(const std::string &data, const std::string
&key)
{
	std::string xorstring = data;
	for (size_t i = 0; i < xorstring.size(); i++)
	{
	  xorstring[i] = data[i] ^ key[i % key.size()];
	}
	return xorstring;
}

std::string SimpleXorCrypt::decrypt(const std::string &data, const std::string
&key)
```

NON-ENCRYPTION USED AS ENCRYPTION

```
{
	return encrypt(data, key);
}
```

The use of a static string (see Finding #**101362** regarding that issue) as the total "encryption" via XOR addition leads to any attacker who can find the key (e.g., through source code disclosure or monitoring messages) being able to encode and decode messages, rendering the value of the "encryption" null.

Impact Rationale:
Given the ability to tamper with messages being sent from the userspace VPN client to the kernelspace helper, the attacker can impersonate the client and execute commands in kernel mode; however, this bug, on its own, only defeats the pseudo-encryption used as a secondary protection on the messages.

Difficulty Rationale:
Only a basic mathematical understanding is required to exploit this finding.

Recommendation

Any data that matters to the enterprise should only be transmitted over the Internet with encryption. Any data at rest that is highly sensitive to disclosure, including both trade secrets and customer PII, should be encrypted at rest, with the encryption key stored apart from the data itself. Encryption should never be written by software engineers; instead, rely on well-tested and supported open-source libraries for encryption, and follow recommendations in their use.

USE OF HARD-CODED CREDENTIALS

ID	101362
Component	Security Best Practices
Severity	**Low**
Impact / Skill Level	Low/Simple
Reference	http://cwe.mitre.org/data/definitions/798.html
Location	backend/windows/windscribe_service/ipc/servicecommunication.h:55

Observation

If hard-coded credentials are embedded in software, the credential is difficult to change at a later date. If the credential becomes publicly known, an attacker can easily break in.
We observed that the software contains hard-coded credentials which it uses for communication with the helper app.

```
#define ENCRYPT_KEY "4WabP[redacted]..."
```

This is the encryption key used by the non-encryption algorithm noted in Finding #101364.

Impact Rationale:
Access to the encryption key allows an attacker to bypass the protection granted by that encryption. In this case, due to a related finding (Finding #**101364**), little protection is granted.

Difficulty Rationale:
Access to a debugger or to the source code discloses the key.

Recommendation

Remove hard-coded credentials and modify functionality that relies on their presence.

NAMED PIPES ARE NOT SECRET

ID	101379
Component	Security Best Practices
Severity	**Low**
Impact / Skill Level	High/Advanced
Reference	https://winaero.com/enable-case-sensitive-mode-windows-10/
Location	backend/windows/windscribe_service/windscribe_service.cpp:279 backend/engine/helper/helper_win.cpp:921

Observation

A named pipe is a file-like object that enables the creating process (called the server in this context, regardless of its purpose) to communicate either unidirectionally or bidirectionally with another process (called the client). While only two processes can attach to a single named pipe at one time, there is no authentication performed on the identity of those processes; hence a process that needs to ensure that messages come from a trustworthy source must employ additional measures to ensure that the process on the other end of a pipe is the expected process.
The Windscribe system service opens a named pipe (thus being the pipe's "server") to communicate with the (client) engine. However, an attacker that can run a malicious process in userspace can either start up before the Windscribe application and attach to that named pipe, or can terminate the Windscribe application and open the named pipe.

Before executing commands from the pipe, the service checks the identity of the process attached to the userspace side of the pipe. Specifically, it checks that the path to the executable that launched the userspace process is case-insensitive string equal to the path to the service executable plus "WindScribeEngine.exe." Windows uses a case-insensitive filesystem by default; however, with support for Windows Subsystem for Linux, Windows added a command to set particular folders as case sensitive. If the check succeeds, the service executes commands using strings sent through the pipe.

This behavior could therefore be exploited in one of two cases:

1. If the Windscribe executables were in a folder that were for some reason marked as case-sensitive, an attacker could create a process with the same path or filename and different capitalization which would be allowed to send commands to the service. (Note that marking a folder as case-sensitive requires elevated permissions, and if the attacker already has elevated permissions, there is no need to exploit Windscribe to gain them.) This is unlikely to happen by accident.
2. If a vulnerability in Windows allowing a process to spoof its executable path is found, then a malicious process could simply provide fraudulent information to pass the service's check.

Since the service executes commands with strings sent through the pipe (without validating those strings), an attacker can use the service's named pipe to execute arbitrary commands with system

privileges.

Impact Rationale:
Successful exploitation could cause a malicious process running as the user to send commands for execution to the Windscribe service, where they would be run with system permissions.

Difficulty Rationale:
An unlikely misconfiguration, or an unknown break in Windows, would be required to exploit this potential vulnerability.

Recommendation

Implement a new interprocess communications method, or (in conjunction with Finding #**101364** regarding encryption) implement effective encryption and key management for both sides of the named pipe to ensure that only processes with system-equivalent access can cause the VPN helper to execute arbitrary commands.

LOCAL DENIAL OF SERVICE

ID	101380
Component	Security Best Practices
Severity	**Low**
Impact / Skill Level	Low/Simple
Reference	https://cwe.mitre.org/data/definitions/412.html https://www.owasp.org/index.php/SameSite https://tools.ietf.org/html/draft-west-first-party-cookies-071 http://cwe.mitre.org/data/definitions/730.html
Location	backend/windows/windscribe_service/windscribe_service.cpp:1102

Observation

If an attacker can prevent a service from fulfilling its purpose or emitting a useful notification to the user, this leads to the user assuming the service is broken.
See Finding #**101379** for background on the checks that the Windscribe service performs on the named pipe.

In the event that a malicious process attaches to the named pipe before the Windscribe userspace process, this would prevent the legitimate userspace process from attaching. This would not allow the attacker to pass malicious commands to the service; however, since the service does not emit an error message nor does it boot the malicious process off the named pipe, the user will assume Windscribe is to blame.

Impact Rationale:
The user will be confused, and the Windscribe service will appear to be nonfunctional.

Difficulty Rationale:
An attacker would have to create a malicious process designed to exploit this behavior and execute it on the user's system.

Recommendation

Implement an "else" case for the "if" check on line 1102 of windscribe_service.cpp, emitting a user-visible error message explaining that a malicious process is interfering with Windscribe.

CUSTOM DEPENDENCY BUILD SYSTEM

ID	101213
Component	Security Best Practices
Severity	**Info**
Impact / Skill Level	Informational/Simple
Reference	N/A
Location	tools/deps/*

Observation

Deviation from conventional practice can result in code that is difficult to modify without introducing errors.

Rather than using package management tools, dependencies in the Windscribe build system are managed by a custom cross-platform source build system that appears to be entirely bespoke. There is significant duplicated-with-small-changes code for each operating system, and it is reasonable to assume that this code would grow by approximately 50% with the addition of Linux OS support (which is anticipated in the next release). The creation, use, and maintenance of this system introduce significant complexities and brittleness into the build process, as well as making it more difficult for developers to begin to be useful when making changes to the software.

Impact Rationale:

While this issue by itself does not have a security impact, it is likely to lead to security impacts due to the complexity of working with the codebase.

Difficulty Rationale:

This issue is apparent to anyone working with the source code.

Recommendation

Replace the dependency system with a simple invocation of the OS-common package manager for each supported OS, and document this in the README in place of running all the dependency system scripts. For instance, on macOS,

```
brew install boost c-ares curl googletest lzo openssl openvpn protobuf qt@5 stunnel
wireguard-go zlib
```

Would handle all the dependencies with the exception of JOM (a clone of nmake), which could either be downloaded by itself or substituted. While there are modifications to many of the source code files, it is not clear that these modifications are necessary and that they cannot also be upstreamed to the main packages (potentially as an optional build parameter).

LACK OF COMMENTS THROUGHOUT CODE

ID	101355
Component	Security Best Practices
Severity	**Info**
Impact / Skill Level	Informational/Simple
Reference	https://www.doxygen.nl/index.html
Location	*

Observation

Comments assist a codebase in being understandable, which enables programmers working on the code to be more efficient and to avoid previously-known mistakes.
The codebase, while generally readable, is almost completely free of comments that explain what parameters are, what inputs should be, what assumptions the coders are making in particular cases, and what reliances exist in the codebase. This is likely to lead to security-impactful bugs in the future as the development expands from a single-developer project to an open-source project, as developers may not completely understand every aspect of the (large) codebase before suggesting changes.

Impact Rationale:
While this issue by itself does not have a security impact, it is likely to lead to security impacts due to the complexity of working with the codebase.

Difficulty Rationale:
This issue is apparent to anyone working with the source code.

Recommendation

As part of a general improvement on code quality, create automatically-parseable (Doxygen) preamble comments for all functions throughout the codebase specifying parameters, return values, and functional assumptions. Within functions, consider adding in-line comments to aid in understanding security-critical or complex operations.

License Review

Windscribe had suffered negative consequences as the result of an open-source license noncompliance issue before the beginning of this engagement; accordingly, they asked Leviathan to evaluate other software used by the Windscribe application for potential noncompliance issues. While Leviathan Security Group is not a law firm and cannot provide legal advice, we are able to evaluate what software is in use and whether the license seems to be complied with in broad strokes. We recommend that Windscribe consult with its counsel with regard to all legal questions.

Threat Analysis

Open-source license litigation can be extremely serious to companies. https://en.wikipedia.org/wiki/Open_source_license_litigation notes a variety of incidents (there are many more that have been resolved before filing a lawsuit), including one where the Free Software Foundation attempted to stop all sales of Cisco products. Last year, a company even filed a $100,000,000 demand for damages in the case of a GPLv2 license violation. (https://www.whitesourcesoftware.com/resources/blog/the-100-million-case-for-open-source-license-compliance/) Even if the case is resolved for less than that amount, as seems likely, there will be significant costs incurred in addition to whatever damages are finally arrived at. Accordingly, license noncompliance is a substantive issue.

After internal discussion, we rated these findings as Medium severity due to the possibility (rather than certainty) of business-ending litigation costs; however, they could also reasonably be rated as High severity due to the amount of money and reputational harm at stake.

Observations

Windscribe appears to be boldly noncompliant with the licenses of **all** open-source software utilized in its product; this assessment found no open-source software included whose license was not currently being broken by Windscribe. This is an unfortunate, but readily remediable, situation; as noted in the findings, creating a place in the application to view the copyright information and licenses of all the included open-source software, along with open sourcing all Windscribe's changes to the libraries, will solve this issue.

Vulnerabilities

LICENSE NONCOMPLIANCE: LZO

ID	101236
Component	License Review
Severity	**Medium**
Impact / Skill Level	Medium/Simple
Reference	https://tldrlegal.com/license/gnu-general-public-license-v2 https://en.wikipedia.org/wiki/Open_source_license_litigation
Location	tools/deps/custom_lzo/B/win32/vc.bat

Observation

Noncompliance with the terms of a software license subjects a noncompliant entity to substantial civil liability.

LZO is licensed under the GPLv2 license (see COPYING in the source tree). As part of the dependency build, one file, **`vc.bat`**, is changed by Windscribe before LZO is built. Since Windscribe is conveying the resulting software to its customers, it must comply with the terms of the software license, which requires making the source code available to the software that it has modified, as well as including the original copyright notice in a way that is visible to end-users.

Impact Rationale:
License noncompliance can result in substantial monetary penalties.

Difficulty Rationale:
Discovery that a library has been included and modified from an available open-source version is usually trivial.

Recommendation

Open source the changes to LZO, and otherwise comply with the requirements of the license. This should include updating the application to include a space where end users can view all OSS libraries used in the application, along with their licenses and copyright notices, per the terms of essentially every open-source license.

LICENSE NONCOMPLIANCE: QT

ID	101238
Component	License Review
Severity	**Medium**
Impact / Skill Level	Medium/Simple
Reference	https://tldrlegal.com/license/gnu-lesser-general-public-license-v3-(lgpl-3) https://www.qt.io/licensing/ https://en.wikipedia.org/wiki/Open_source_license_litigation
Location	tools/deps/custom_qt/source_changes.json

Observation

Noncompliance with the terms of a software license subjects a noncompliant entity to substantial civil liability.

QT is licensed under the GPLv3 or LGPLv3 license. As part of the dependency build, two files are changed by Windscribe before QT is built. Since Windscribe is conveying the resulting software to its customers, it must comply with the terms of the software license, which requires making the source code available to the software that it has modified, as well as including the original copyright notice in a way that is visible to end-users.

Impact Rationale:
License noncompliance can result in substantial monetary penalties.

Difficulty Rationale:
Discovery that a library has been included and modified from an available open-source version is usually trivial.

Recommendation

Open source the changes to QT, and otherwise comply with the requirements of the license. This should include updating the application to include a space where end users can view all OSS libraries used in the application, along with their licenses and copyright notices, per the terms of essentially every open-source license.

LICENSE NONCOMPLIANCE: STUNNEL

ID	101239
Component	License Review
Severity	**Medium**
Impact / Skill Level	Medium/Simple
Reference	https://tldrlegal.com/license/gnu-general-public-license-v2 https://www.stunnel.org/COPYING.html https://en.wikipedia.org/wiki/Open_source_license_litigation
Location	tools/deps/custom_stunnel/src/vc.mak

Observation

Noncompliance with the terms of a software license subjects a noncompliant entity to substantial civil liability.

Stunnel is licensed under the GPLv3 license. As part of the dependency build, one file, **`vc.mak`**, is changed by Windscribe before stunnel is built. Since Windscribe is conveying the resulting software to its customers, it must comply with the terms of the software license, which requires making the source code available to the software that it has modified, as well as including the original copyright notice in a way that is visible to end-users.

Impact Rationale:
License noncompliance can result in substantial monetary penalties.

Difficulty Rationale:
Discovery that a library has been included and modified from an available open-source version is usually trivial.

Recommendation

Open-source the changes to stunnel, and otherwise comply with the requirements of the license. This should include updating the application to include a space where end users can view all OSS libraries used in the application, along with their licenses and copyright notices, per the terms of essentially every open-source license.

LICENSE NONCOMPLIANCE: C-ARES

ID	101240
Component	License Review
Severity	**Medium**
Impact / Skill Level	Medium/Simple
Reference	https://tldrlegal.com/license/mit-license https://c-ares.org/license.html https://en.wikipedia.org/wiki/Open_source_license_litigation
Location	tools/deps/install_cares.py

Observation

Noncompliance with the terms of a software license subjects a noncompliant entity to substantial civil liability.

c-ares is licensed under the MIT license. Since Windscribe is conveying the resulting software to its customers, it must comply with the terms of the software license, which requires including the original copyright notice in a way that is visible to end-users.

Impact Rationale:

License noncompliance can result in substantial monetary penalties.

Difficulty Rationale:

Discovery that a library has been included is usually trivial.

Recommendation

Comply with the requirements of the license. This should include updating the application to include a space where end users can view all OSS libraries used in the application, along with their licenses and copyright notices, per the terms of essentially every open-source license.

LICENSE NONCOMPLIANCE: OPENVPN, TAP-WIN32/TAP-WIN64, WINDOWS DDK, NSIS, OPENSSL, SSLEAY

ID	101241
Component	License Review
Severity	**Medium**
Impact / Skill Level	Medium/Simple
Reference	https://github.com/OpenVPN/openvpn/blob/master/COPYING https://tldrlegal.com/license/gnu-general-public-license-v2 https://tldrlegal.com/license/zlib-libpng-license-(zlib) https://tldrlegal.com/license/bsd-3-clause-license-(revised) https://en.wikipedia.org/wiki/Open_source_license_litigation
Location	tools/deps/custom_openvpn

Observation

Noncompliance with the terms of a software license subjects a noncompliant entity to substantial civil liability.
OpenVPN is licensed under the GPLv2. As part of the dependency build, a variety of files are changed by Windscribe before OpenSSL is built. Since Windscribe is conveying the resulting software to its customers, it must comply with the terms of the software license, which requires making the source code available to the software that it has modified, as well as including the original copyright notice in a way that is visible to end-users.

In addition, OpenVPN itself packages a variety of its dependencies which are themselves subject to various open-source licenses. These dependencies are as follows:
LZO: GPL + OpenSSL Exception
TAP-Win32/TAP-Win64: GPLv2
NSIS: zlib/libpng License (notice)
OpenSSL: OpenSSL License + SSLeay License (BSD-style)

Accordingly, noncompliance with the OpenVPN license entails noncompliance with several other open-source licenses as well, each library having an independent cause of action which may subject Windscribe to liability.

Impact Rationale:
License noncompliance can result in substantial monetary penalties.

Difficulty Rationale:
Discovery that a library has been included is usually trivial.

Recommendation

Open source the changes to OpenVPN, and otherwise comply with the requirements of the license. This should include updating the application to include a space where end users can view all OSS libraries

LICENSE NONCOMPLIANCE: OPENVPN, TAP-WIN32/TAP-WIN64, WINDOWS DDK, NSIS, OPENSSL, SSLEAY

used in the application, along with their licenses and copyright notices, per the terms of essentially every open-source license.

LICENSE NONCOMPLIANCE: WIREGUARD

ID	101284
Component	License Review
Severity	**Medium**
Impact / Skill Level	Medium/Simple
Reference	https://tldrlegal.com/license/mit-license https://github.com/WireGuard/wireguard-go/blob/master/LICENSE https://en.wikipedia.org/wiki/Open_source_license_litigation
Location	tools/deps/install_wireguard.py

Observation

Noncompliance with the terms of a software license subjects a noncompliant entity to substantial civil liability.
Wireguard-go is licensed under the MIT license. Since Windscribe is conveying it to its customers, it must comply with the terms of the software license, which requires including the original copyright notice in a way that is visible to end-users.

Impact Rationale:
License noncompliance can result in substantial monetary penalties.

Difficulty Rationale:
Discovery that a library has been included and modified from an available open-source version is usually trivial.

Recommendation

Comply with the requirements of the license. This should include updating the application to include a space where end users can view all OSS libraries used in the application, along with their licenses and copyright notices, per the terms of essentially every open-source license.

Appendix A – Technical Services

Leviathan's Technical Services group brings deep technical knowledge to your security needs. Our portfolio of services includes software and hardware evaluation, penetration testing, red team testing, incident response, and reverse engineering. Our goal is to provide your organization with the security expertise necessary to realize your goals.

SOFTWARE EVALUATION We provide assessments of application, system, and mobile code, drawing on our employees' decades of experience in developing and securing a wide variety of software. Our work includes design and architecture reviews, data flow and threat modeling, and code analysis using targeted fuzzing to find exploitable issues.

HARDWARE EVALUATION We evaluate new hardware devices ranging from novel microprocessor designs, embedded systems, mobile devices, and consumer-facing end products, to core networking equipment that powers internet backbones.

PENETRATION & RED TEAM TESTING We perform high-end penetration tests that mimic the work of sophisticated attackers. We follow a formal penetration testing methodology that emphasizes repeatable, actionable results that give your team an understanding of the overall security posture of your organization as well as the details of discovered vulnerabilities.

SOURCE CODE-ASSISTED SECURITY EVALUATIONS We conduct security evaluations and penetration tests based on our code-assisted methodology, allowing us to find deeper vulnerabilities, logic flaws, and fuzzing targets than a black-box test would reveal. This methodology gives your team a stronger assurance that the most significant security-impacting flaws have been found, allowing your team to address them.

INCIDENT RESPONSE & FORENSICS We respond to our customers' security incidents by providing forensics, malware analysis, root cause analysis, and recommendations for how to prevent similar incidents in the future.

REVERSE ENGINEERING We assist clients with reverse engineering efforts. We provide expertise in investigations and litigation by acting as experts in cases of suspected intellectual property theft.

Appendix B – Risk and Advisory Services

Leviathan's Retained Services group is a supplement to an organization's security and risk management capability. We offer a pragmatic information security approach that respects our clients' appetites for security process and program work. We provide access to industry leading experts with a broad set of security and risk management skills, which gives our clients the ability to have deep technical knowledge, security leadership, and incident response capabilities when they are needed.

INFORMATION SECURITY STRATEGY DEVELOPMENT We partner with boards, directors, and senior executives to shape your enterprise's overall approach to meeting information security requirements consistently across an entire organization.

ENTERPRISE RISK ASSESSMENT We develop an information asset-centric view of an organization's risk that provides insight to your organization's Enterprise Risk Management capability. This service can be leveraged with annual updates, to account for your organization's changing operations, needs, and priorities.

PRIVACY & SECURITY PROGRAM EVALUATION We evaluate your organization's existing security program to give you information on compliance with external standards, such as ISO 27000 series, NIST CSF, HIPAA, or PCI-DSS. This is often most useful before a compliance event or audit and helps to drive the next phase of growth for your Security and Risk Management programs.

VENDOR RISK ASSESSMENT We assess the risk that prospective vendors bring to your organization. Our assessment framework is compatible with legislative, regulatory, and industry requirements, and helps you to make informed decisions about which vendors to hire, and when to reassess them to ensure your ongoing supply chain security.

NATIONAL & INTERNATIONAL SECURITY POLICY In 2014, we launched a public policy research and analysis service that examines the business implications of privacy and security laws and regulations worldwide. We provide an independent view of macro-scale issues related to the impact of globalization on information assets.

M&A/INVESTMENT SECURITY DUE DILIGENCE We evaluate the cybersecurity risk associated with a prospective investment or acquisition and find critical security issues before they derail a deal.

LAW FIRM SECURITY SERVICES We work with law firms as advisors, to address security incidents and proactively work to protect client confidences, defend privileged information, and ensure that conflicts do not compromise client positions. We also work in partnership with law firms to respond to their clients' security needs, including in the role of office and testifying expert witnesses.

SAAS AND CLOUD INITIATIVE EVALUATION We give objective reviews of the realistic threats your organization faces both by moving to cloud solutions and by using non-cloud infrastructure. Our employees have written or contributed to many of the major industry standards around cloud security, which allows their expertise to inform your decision-making processes.