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

Pentest & Audit Report Jigsaw Outline 09.-12.2018

Cure53, Dr.-Ing. M. Heiderich, MSc. N. Krein, BSc. T. C. “Filedescriptor” Hong, BSc. J. Hector, MSc. N. Kobeissi, Dipl.-Ing. A. Inführ

Index

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

Introduction

"Journalists need safe access to information to research issues, communicate with sources, and report the news. Outline lets news organizations easily provide their network safer access to the open internet."

From https://www.getoutline.org/en/home

This report documents the results of a security assessment targeting the Jigsaw Outline VPN tools. The project was carried out by Cure53 in September 2018 and yielded twelve security-relevant findings.

This Cure53 assessment encompassed both a penetration test and a source code audit. It had a clear scope consisting of three distinct components, namely the *Outline* server, the *Outline* clients and the *Outline Manager* application. As for the timeline, nineteen days were invested into the completion of this test and audit, with the core budget dedicated to security work being accompanied by the tasks around communications, coordination and documentation. The majority of the work took place in Calendar Week 39 (CW39) in September 2018.

Moving on to the personnel resources, it needs to be noted that six members of the Cure53 were involved in this assessment of the Jigsaw Outline compound. The methodology chosen for the project was a white-box approach, especially since most of the relevant code is available as open-source. To ensure best possible coverage, Cure53 conducted both a pentest and an SCA. Overall, the project proceeded smoothly and adhered to the predetermined schedule. The communications with the maintaining team over at Jigsaw was done via email since the test went fluently and no live-chat tool was necessary.

As already noted, Cure53 managed to unveil twelve issues but only four among them were categorized as actual vulnerabilities. The remaining array of eight findings encompassed general weaknesses with generally lower or even negligible impact. However, in the first category of vulnerabilities, two items received "*High"*-severity rankings, as they could signify considerable damage and risks for the Jigsaw Outline scope. All issued had been live-reported to the Jigsaw team while the tests were still ongoing. This means that work on addressing the bugs with fixes could have begun before the finalization of this report.

In the following sections, the report first describes the scope and then discusses the findings on a case-by-case basis, furnishing mitigation advice when applicable. Lastly, the document will close with the conclusion, which sheds light on the general outcome of the test and audit, expressing also the verdict that Cure53 has reached with reference to the security posture o the Jigsaw Outline VPN software compound.

***Note:** This report was amended on December 11th 2018 to reflect the state after the fix verification performed by Cure53. Each ticket was extended with a note reporting on the status of the fix and its respective verification.*

Scope

- **Outline VPN Platform**
  - Outline VPN server software
    - https://github.com/Jigsaw-Code/outline-server
  - Outline SS Server
    - https://github.com/Jigsaw-Code/outline-ss-server
  - Outline VPN client apps
    - https://github.com/Jigsaw-Code/outline-client
  - Outline VPN *Manager* Electron application (with the same repo as the server)
    - https://github.com/Jigsaw-Code/outline-server#components
  - Available Binaries
    - https://github.com/Jigsaw-Code/outline-server/releases

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. *JIG-01-001*) for the purpose of facilitating any future follow-up correspondence.

JIG-01-002 Server: Sensitive info-leak due to weak file permissions *(Medium)*

It was discovered that the *Outline* server component stores sensitive information on the filesystem. This has been proven for the server API URL and the *shadowbox* credentials. Since the files are marked as world-readable, any user on the system can access their contents. To exploit this vulnerability, either a compromised user account or a file disclosure vulnerability need to be present on a server that runs *shadowbox*.

File Permissions:
```
$ cd ~/shadowbox/
$ ls -la
```

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

```
total 16
drwxrwxr-x  3 user user 4096 Sep 25 13:45 .
drwxr-xr-x 11 user user 4096 Sep 26 08:35 ..
-rw-rw-r--  1 user user  133 Sep 17 16:08 access.txt

$ cd persisted-state/
$ ls -la
total 28
drwxrwxr-x 2 user user 4096 Sep 25 15:51 .
drwxrwxr-x 3 user user 4096 Sep 25 13:45 ..
-rw-r--r-- 1 root root  344 Sep 24 14:04 shadowbox_config.json
[...]
```

The entire security of the server API endpoint is based on the fact that the endpoint's *name* is randomly chosen. Since the *access.txt* is world-readable, any user on the system can read it, thereby gaining full control over the *management* interface.

It is recommended to remove access permissions for all users with the exception of the owner. Inspecting the running processes on the test system revealed that the server is running as *root* and should thus still be able to read *shadowbox_config.json*.

***Note:** This issue was fixed by the maintainers and verified fixed by Cure53 in December 2018. The issue is no longer present in the described form. Read permissions have been adjusted correctly using umask[1].*

JIG-01-004 Client: *DigitalOcean* takeover via XSS on callback of *Manager* (*High*)

It was found that XSS exists on the callback endpoint of the *localhost* server spawned by *Outline Manager* for *DigitalOcean* OAuth authorization. When a user is installing Outline server on DigitalOcean, *Outline Manager* temporarily spawns a HTTP server on *localhost*. The user is then redirected to DigitalOcean's website and asked if they want to authorize access to *Outline Manager*. Finally, the *access* token is sent to the whitelisted callback URL (i.e. *localhost*). Since a XSS can access DOM objects on the same executing origin, this allows to steal the *access* token of a user's DigitalOcean account with full access by retrieving the final redirected URL.

To exploit this vulnerability, the attacker needs to send a crafted URL to the victim, and the victim needs to open it with a computer that is running *Outline Manager* and is awaiting a response from DigitalOcean.

Affected File:
outline-server/src/server_manager/electron_app/digitalocean_oauth.ts

[1] https://github.com/Jigsaw-Code/outline-server/pull/311

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

Affected Code:

```
var paramsStr = location.hash.substr(1);
var params = splitParams(paramsStr);
var form = document.getElementById("form");
var targetUrl = params["state"];
form.setAttribute("action", targetUrl);
document.getElementById("params").setAttribute("value", paramsStr);
form.submit();
```

Steps to Reproduce:

1. Open *Outline Manager* and select *DigitalOcean;*
2. Navigate to http://localhost:55189/#state=javascript:alert(document.domain);
3. An alert box will pop up.

It is recommended to validate the *state* parameter to be a HTTP(s) URL.

***Note:** This issue was fixed by the maintainers and verified fixed by Cure53 in December 2018. The issue is no longer present in the described form. The parameter is now properly encoded and attached to a static relative URL[2].*

JIG-01-011 AWS: DOM XSS via invite URL on *Invite* page *(Low)*

The *Outline* application uses a public *AWS* HTML page to share server information with clients. It was discovered that this HTML page suffers from a DOM XSS vulnerability, which allows to execute JavaScript in the context of the *invite.html* page.

The *AWS* HTML page parses the invitation link specified in the location hash. The deployed JavaScript only checks if the specified URL starts with *ss://* but does not encode the URL before including it in the DOM. By specifying a HTML structure after the protocol, the HTML tag gets parsed. The defined JavaScript is then executed.

To exploit this issue, a specially crafted invite URL needs to be sent to a victim, who is not only using the Outline client but is also expecting an invitation URL to be able to connect to a new Outline VPN server.

Proof-of-Concept:

*https://s3.amazonaws.com/outline-vpn/invite.html#/en/invite/ss://**<img src=x onerror=alert(location)>***

[2] https://github.com/Jigsaw-Code/outline-server/pull/285

Rendered HTML:
```
<li>Copy your access key: <span class="code">ss://<img src="x" onerror="alert(location)"></span></li>
```

Although it is necessary to share a malicious URL with a victim, it is recommended to URL-encode HTML characters specified in the *invite* link before it is being placed in the DOM. This will ensure that no attacker-controlled HTML tag is parsed by the server.

***Note:** This issue was fixed by the maintainers and verified fixed by Cure53 in December 2018. The issue is no longer present in the described form. The parameter is now properly encoded[3].*

JIG-01-012 Client: DNS leaks on Win 8 & 10 due to unexpected behavior (*High*)

It was found that when one uses Jigsaw *Outline* on Windows 8 and 10, it is possible to experience DNS leaks. Despite *Outline* installing a *TAP* driver with the highest network adapter metric priority, the aforementioned versions of Windows issue DNS requests to all available network interfaces[4]. This happens regardless of the network metric configurations. To be more specific, Windows 8 has introduced a feature called "*Smart Multi-Homed Name Resolution*".

This item issues DNS requests in parallel to all available network adapters and picks the fastest response. The issue can simply be reproduced by visiting https://ipleak.net while using the *Outline*'s Windows client on a Windows 10 or 8 machine. This allows network onlookers to unveil websites that other users are visiting, and even hijack the DNS requests to serve unintended contents.

Although Windows provides means to disable the feature (via *Registry* and *Local Group Policy*), upon closer investigation it was discovered that even after disabling the feature, minor DNS leaks remain. This means that fixing this issue is not a trivial task.

Any Windows 8.x/10 user who has DNS servers configured on any network interface other than *Outline's TAP* driver are affected by this issue. It should be noted that even if those DNS servers are trusted (e.g. Public DNS servers), a local network attacker can still intercept and modify DNS requests since they are going through unencrypted UDP traffic from their original network interfaces.

Due to the unexpected DNS behavior on Windows, it is recommended to hook the Windows DNS resolve to allow only DNS requests originated from *Outline*'s *TAP* driver.

[3] https://github.com/Jigsaw-Code/outline-website/pull/100
[4] https://medium.com/@ValdikSS/beware-of-windows-10-dns-resolver-and-dns-leaks-5bc5bfb4e3f1

OpenVPN uses exactly this method to prevent unexpected DNS leaks with the *--block-outside-dns* flag[5] and this can be followed here.

***Note:** This issue was fixed by the maintainers and verified fixed by Cure53 in December 2018. The issue is no longer present in the described form. The newly deployed and executed binary blocks all DNS requests on devices other than the VPN connection[6].*

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.

JIG-01-001 Design: Unrestricted access to *OutlineService* can disable VPN *(Info)*

A service installed on Windows is responsible for adding the essential routes for establishing the VPN connection. This service is used by the client application through named pipes. However, any authenticated user on the system is able to connect to said named pipe and can issue arbitrary commands, for example s/he can disable the VPN entirely.

A Proof-of-Concept (PoC) *node.js* script below connects to the named pipe and sends the command to disable the VPN routes.

PoC *node.js* script:
```
const net = require('net');

const SERVICE_PIPE_NAME = 'OutlineServicePipe';
const SERVICE_PIPE_PATH = '\\\\.\\pipe\\';

const request = {action: "resetRouting", parameters: {}};

ipccon = net.createConnection(`${SERVICE_PIPE_PATH}${SERVICE_PIPE_NAME}`, () =>
{
        console.log('Pipe connected');
        try {
                const msg = JSON.stringify(request);
                ipccon.write(msg );
        } catch (e) {
                console.log(`error: ${e.message}`);
        }
```

[5] https://github.com/ValdikSS/openvpn-fix-dns-leak-plugin
[6] https://github.com/Jigsaw-Code/outline-client/pull/394

```
});

ipccon.on('error', (err) => {
	throw err;
});

ipccon.on('data', (data) => {
	console.log(data.toString());
	ipccon.destroy();
});
```

Therefore, even a regular user without any administrative privileges is able to disable the VPN for every other user on the system and can also easily deanonymize others. Additionally, the service can be abused to route traffic through an attacker-controlled server.

In a multi-user scenario, fixing this kind of issue should be considered rather difficult. One way would be to implement some form of access-control that depends on the state of the VPN. One approach would be to only have *administrator/owner* user being allowed to disable the VPN once it is activated.

This ticket is meant to inform the developers about the potential risk; fixing this issue depends on the threat model and whether or not malicious local users are part of it.

Note: *This issue was flagged as wontfix given that a shared computer or a computer with potentially malicious users is not part of the threat model.*

JIG-01-003 Server: Default server installer adds user to the *docker* group (*Info*)

Outline provides an installer script that allows to easily set up a *shadowbox* server. One of the steps in the installation is to add the user executing the script to the *docker* group. However, this is considered a bad practice, since being assigned to the *docker* group can be abused for privilege escalation[7]. To exploit this vulnerability and conduct privilege escalation, an attacker needs to compromise the user account that was used to install *shadowbox* on the server.

Affected File:
outline-server/src/server_manager/install_scripts/install_server.sh

Affected Code:

```
function verify_docker_permissions() {
  if user_in_docker_group; then
```

[7] https://fosterelli.co/privilege-escalation-via-docker.html

```
    return 0
  fi
  log_error "FAILED"
  local readonly PROMPT="> It seems like you may not have permission to run
Docker. To solve this, we will attempt to add your user to the docker group by
running 'sudo usermod -a -G docker $USER'. Would you like to proceed? [Y/n] "
  if ! confirm "$PROMPT"; then
    exit 0
  fi
  if run_step "Adding $USER to docker group" add_user_to_docker_group; then
[...]
function add_user_to_docker_group() {
  sudo usermod -a -G docker $USER
}
```

Although there is a prompt asking whether or not the user should be added to the group, the user is required to be part of the *docker* group, or the installer will not work. It is recommended to cease the addition of users to the *docker* group and instead rely on *sudo*. Otherwise, the server is exposed to an additional risk and a potential local attacker can leverage this flaw to take over the entire system.

Note: *This issue was fixed by the maintainers and verified fixed by Cure53 in December 2018. The issue is no longer present in the described form. All code that adds users to the docker group was removed, the software instead relies on sudo only*[8].

JIG-01-005 Client: *BrowserWindow* missing security flags in Electron (*Info*)

It was found that both *Outline* server and *Outline* client use *BrowserWindow* of Electron without the *contextIsolation* and *nodeIntegration* flags enabled. While normally being able to navigate to an external page on the BrowserWindow could also lead to RCE, this is not the case here since navigation is trapped with the *will-navigate* event handler. However, this could still lead to RCE if the application becomes vulnerable to XSS.

Affected Files:
outline-server/src/server_manager/electron_app/index.ts
outline-client/electron/index.ts

Affected Codes:
```
function createMainWindow() {
  const win = new electron.BrowserWindow({
    width: 600,
    height: 768,
    resizable: false,
```

[8] https://github.com/Jigsaw-Code/outline-server/pull/312

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

```
    icon: path.join(__dirname, 'web_app', 'ui_components', 'icons',
'launcher.png'),
    webPreferences: {
      nodeIntegration: false,
      preload: path.join(__dirname, 'preload.js'),
      nativeWindowOpen: true,
      webviewTag: false
    }
  });
[...]
function createWindow(connectionAtShutdown?: SerializableConnection) {
  // Create the browser window.
  mainWindow = new BrowserWindow({width: 360, height: 640, resizable: false,
icon: iconPath});
```

It is recommended to set the aforementioned security-relevant flags.

***Note:** This issue was flagged as wontfix given that no user-controlled code is supposed to reach the unprotected Electron windows.*

JIG-01-006 Design: Static files hosted on AWS shared S3 domain *(Low)*

It was found that most external links on both *Outline Manager* and *Outline* client point to the publicly shared S3 domain, which has *outline-vpn* as the namespace. For example, the *Help Center* points to https://s3.amazonaws.com/outline-vpn/index.html#/en/support. The problem here is that anyone can create their own bucket and upload any HTML file onto the same domain. This allows an attacker to spoof the hosted files. In the following PoC, navigating to the specified item will show *s3.amazonaws.com/outline-vpn/index.html* in the address bar with spoofed contents.

PoC:
http://s3.amazonaws.com/pentest.org/outline.html

Code:
```
<script>history.replaceState(1, 1, '/outline-vpn/index.html')</script>
<h1>Visit Cure53.de</h1>
```

It is recommended to use the S3 subdomain instead (i.e. http://outline-vpn.s3.amazonaws.com/index.html).

***Note:** This issue was flagged as wontfix given that an attacker who can successfully invite users can do worse already, and the shared S3 domain is used to help bypassing censorship mechanisms.*

JIG-01-007 Client: *shell.openExternal* missing URL validation in Electron (*Info*)

It was noticed that both *Outline* server and *Outline* client capture the *navigate* event and call *shell.openExternal* in Electron. This could lead to potential RCE if an attacker can somehow make the applications navigate to a local files (e.g. HTML injections).

Affected Code:

```
const handleNavigation = (event: Event, url: string) => {
    shell.openExternal(url);
    event.preventDefault();
  };
  win.webContents.on('will-navigate', (event: Event, url: string) => {
    handleNavigation(event, url);
});
```

It is recommended to check if the URL starts with http(s) before calling *shell.openExternal*.

***Note:** This issue was fixed by the maintainers and verified fixed by Cure53 in December 2018. The issue is no longer present in the described form. The URL is now properly parsed and the protocol part is validated afterwards.*

JIG-01-008 Design: *DigitalOcean* auth model compromisable via Phishing (*Low*)

It was noticed that *Outline Manager* registers *DigitalOcean* applications to enable the authorization of the *DigitalOcean* to *Outline* using OAuth. This is not a recommended approach as the application's owner can be compromised and their callback URL can be changed to a malicious one. In such a scenario, an attacker could steal *access* tokens of *Outline* users and gain full account access.

Note however that this vulnerability needs to be combined with a classic Phishing attack to be exploitable, hence the assigned rating as a miscellaneous issue with rater low severity. A revised approach would be to instruct users to generate their own, personal *access* tokens[9] instead of authorizing a centralized third-party application.

***Note:** This issue has an identified fix, that requires the involvement of DigitalOcean, whom the maintainer team has already reached out to.*

[9] https://www.digitalocean.com/docs/api/create-personal-access-token/

JIG-01-009 Design: Clipboard data can overwrite imported server info *(Low)*

During the assessment of the Android client, it was discovered that the client is always parsing the clipboard for potential *Outline* server information. In case the application is invoked via the custom *ss://* protocol handler, the received server information is parsed. However, as soon as the received information is shown, the clipboard is parsed as well. In case it contains *Outline* server information, the clipboard data will overwrite the latest server information. These could be potentially abused to trick a user into importing malicious server information.

To be able to exploit this issue, a victim needs to visit either a malicious website or have a malicious application installed, which pushes Outline server information into the clipboard. Afterwards, the victim has to import a new Outline VPN server, without overwriting the clipboard data or else the attack will fail.

Custom protocol handler:
```
ss://Y2hhY2hhMjAtaWV0Zi1wb2x5MTMwNTp3dWxTTXhHcEV4SUM=@8.8.8.8:17944/?outline=1
```

Clipboard:
```
ss://Y2hhY2hhMjAtaWV0Zi1wb2x5MTMwNTp3dWxTTXhHcEV4SUM=@9.9.9.9:17944/?outline=1
```

Imported server:
```
9.9.9.9
```

It is recommended to stop the parsing of the clipboard at every occasion when the application has been invoked via the custom protocol handler. This ensures that no data is overwritten during the import process.

Note: *This issue was fixed by the maintainers and verified fixed by Cure53 in December 2018. The issue is no longer present in the described form. An additional check was added to test whether the dialog is already opened and, if that is the case, the clipboard data is ignored*[10]*.*

[10] https://github.com/Jigsaw-Code/outline-client/pull/384

JIG-01-010 Client: Discrepancy between client state and actual VPN state *(Low)*

It was discovered that the *Outline* Windows client app and the actual VPN state can be de-synced through issuing commands directly to the *Outline* service. This can lead to a false impression of being connected to a VPN, while in reality the VPN is actually disconnected.

If a user connects to a given VPN server through the app, the changes in the application indicate that a connection has been established. When this connection is terminated outside of the client app, for example when using JIG-01-001, the app will still indicate that the VPN is connected.

It is recommended to implement a mechanism for the service to communicate the change of state to the app. This will ascertain that the current VPN status is displayed correctly.

Note: *This issue was fixed by the maintainers and verified fixed by Cure53 in December 2018. The issue is no longer present in the described form. Now the software checks for a connection between client and service such that the service can notify the client upon changes*[11]*.*

[11] https://github.com/Jigsaw-Code/outline-client/pull/409

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

Conclusions

Note: *The paragraphs below contain the original conclusion that describes the state of security and privacy before the fix verification, successfully conducted by Cure53 in December 2018. In the current state of affairs, the majority of issues have been fixed and the fixes have been verified successfully.*

The results of this Cure53 assessment of the security posture found on the Jigsaw Outline compound are rather positive. By employing the white-box methodology to test the scope over the course of nineteen days in September 2018, six members of the Cure53 team involved in this project can conclude that the Jigsaw Outline sub-components in scope generally held well to the external, security-centered scrutiny.

Even though a total number of twelve issue may appear high at first glance, the actual level of severities and risks ascribed to these discoveries need to be considered. Taking that into account, one can clearly see that only a handful of issues (i.e. four) signify actual vulnerabilities. The issues with real-life impact of potentially harming the users of Outline were few and far between on the scope.

In terms of the cryptography, Cure53 wishes to underscore that the coverage was fairly straightforward as both server and client closely follow the Shadowsocks specifications. Not only do they not deviate from the standards, but in fact they restrict the premise further and end up only allowing one cipher suite. This is clearly beneficial for the overall integrity of the Jigsaw Outline project.

The issues with the most concerning severities are documented under JIG-01-004 and JIG-01-012. The first of the two describes an XSS vulnerability inside the *callback* manager of the OAuth implementation for *DigitalOcean*. In a worst-case scenario this not only leads to simple Cross-Site Scripting (XSS), but may signal an account takeover effectively executed against the *Outline* user's *DigitalOcean* accounts. As for the latter problem, this concerns privacy aspects. JIG-01-012 describes a DNS leak issue that was discovered when Windows 8 or 10 have been used as platforms to run *Outline* on. This is due to Windows going through a few architectural changes that make it a lot harder to prevent DNS leaks, even when an additional TAP interface is used. Since Outline explicitly states that it "*protects your privacy from network onlookers in the country from where you're accessing the internet, such as your local ISP or national gateway*", Cure53 strongly believes that Outline needs to take the necessary steps and prevent this issue.

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

Moving on to less-concerning threats, some of the issues indicate problems that arise on default installations. One example is that permissions on sensitive files that give insight on the backend API have been incorrectly set (see JIG-01-002). Another risk is connected to the potential privilege escalation problem happening via the default installer for Shadowbox (see JIG-01-003). What is more, the Jigsaw Outline maintainers should dedicate more efforts into modern and holistic hardening features for the Electron application, which were missing, as well as improve input-validation mechanisms (see JIG-01-008 and JIG-01-007).

All in all, Cure53 did not manage to uncover too many highly critical security-issues. Especially the Cordova client applications left a lasting positive impression, as it has been extremely difficult to find any problems in this realm. Upon fixing the described vulnerabilities, the robustness of Outline as a VPN option will be even better, offering a considerably limited attack surface.

In light of the findings and the results of the fix verification, Cure53 can ascertain that the Jigsaw Outline project is on the right track and presents itself as a strong compound with a very good security posture.

Cure53 would like to thank Santiago Andrigo, Vinicius Fortuna and Trevor Johnston from the Jigsaw team for their project coordination, support and assistance, both before and during this assignment.