UnknownVPN Write-Up
CyberSecurity and Risk Analysis of UnknownVPN.net

L33TSTR33T Cybersecurity Collective

01/05/2021

DISCLAIMER: This writeup and the audit were completed following guidelines set at https://cheatsheetseries.owasp.org/cheatsheets/Vulnerability_Disclosure_Cheat_Sheet.html

Table of Contents

Synopsis

It will be noted that for some vulnerabilities within the writeup, imagery and/or code samples to reproduce will be missing. This is due to miscommunication between the team and UnknownVPN and will be impossible to reproduce. We will try to supplement with whatever data we do have and explain with words as best as possible, but due to the lack of availability of the site with which we tested our ability to explain will be hampered. In other words UnknownVPN changed the whole website mid-pentest.

Also, during contact between the team and the company, there were disputes over the actual impact of the vulnerabilities within. We insisted on following CVSS standards when it came to scoring and rewarding bounties, but during disclosure this was rejected and most of these vulnerabilities were considered redundant and unimportant to company infrastructure. You may find within that a lot of the issues covered impact the security of the site in a large enough way that the practice used by the company would be considered irresponsible.

Lastly, most, if not all of the vulnerabilities below have been “patched” and are not reproducible; the Company deleted all relevant source code and published a pure html version of the website in its place.

The last section of the writeup will include vulnerabilities that fit in one of the following:

- Have a CVSS Score of 0.
- Are potential vulnerabilities that we can not confirm.
- Other security issues with the service that we believe should be acknowledged.

Low Impact Vulnerabilities

This section will describe the Low Impact Vulnerabilities found within UnknownVPN and UnknownVPN related software. By definition, a Low Impact Vulnerability is one with a calculated CVSS score of 0.1-3.9. The CVSS vector string will be provided with all vulnerabilities.

Failure to Invalidate Session Cookie After Logout

CVSS:3.1/AV:N/AC:L/PR:H/UI:N/S:U/C:L/I:L/A:N CVSS Score: 3.8 CWE-ID: CWE-284

When testing for another vulnerability, I discovered that cookie data persists in every request you send after you log out. If chained with XSS, a user's hash, name, email address and more can be compromised due to cookies not clearing when using the logout button within the User's dashboard.

While it is low impact, the general end user would not feel secure knowing potentially identifying information is capable of being leaked even while being logged out.

Solution:
Delete all cookies unrelated to cloudflare when logging out. Also, set a timer on how long someone's session can last, and lastly, find a technical workaround to the wdData array within cookies that holds said information. It's presence is insecure and serves no purpose, as it is not a protection against the CVSS 9.6 account takeover vulnerability.

Medium Impact Vulnerabilities

This section will describe the Medium Impact Vulnerabilities found within UnknownVPN and UnknownVPN related software. By definition, a Medium Impact Vulnerability is one with a calculated CVSS score of 4.0-6.9. The CVSS vector string will be provided with all vulnerabilities.

Information Disclosure via Version.txt

CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:N/I:L/A:N CVSS Score:4.7 CWE-ID: CWE-201

There is a file stored on http://api.unknownvpn.net/version.txt hosting information that could be manipulated to impact the end user. The file's contents at the time of writing include:

```
1.6.0.0|https://github-production-release-asset-2e65be.s3.amazonaws.com/304969774/56fafb00-10a1-11eb-8617-a0471259c115?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAIWNJYAX4CSVEH53A%2F20201017%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20201017T215237Z&X-Amz-Expires=300&X-Amz-Signature=829e2aef8f55448acd0539914f65eb825f4b8f1c8a88298513d2c0fd0ae3d1b2&X-Amz-SignedHeaders=host&actor_id=70519085&key_id=0&repo_id=304969774&response-content-disposition=attachment%3B%20filename%3DUnknownVPN-Client-Update-1.6.zip&response-content-type=application%2Foctet-stream
```

The Company claims that the file contains no sensitive information. However, if this data were to be edited via another vulnerability, it would compromise the security of the end user. This is due to the information within the file being executed within the UnknownVPN Client software.

Impact:
It can be argued that the impact of this vulnerability is minor, but due to its potential as a vector in an exploit chain and the general lack of security within the site, it earned a CVSS score of 4.7 and therefore is classed as a medium impact vulnerability.

Solution:
The most optimal solution here, given the contents of the file, would be to implement git version control directly into the application. This would serve the same purpose as the version.txt, while removing the potential for code to be executed on a user's system.

Modification of Assumed-Immutable Data

CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:N/E:P/RL:O/RC:C CVSS Score: 5.9 CWE-ID: CWE-471

During the purchase of a token for the service, you are capable of manipulating the price of the overall value of your cart due to improper controls within /view/front/controller.php. The impact of this vulnerability can be maximized due to you being able to buy as many tokens as you want, therefore devaluing the product. The minimum that you may spend during a paypal transaction is USD$0.01; when coupled with the impact of a virtually unlimited cart, you can create thousands of dollars of impact for the price of 1 cent.

There are many ways to replicate this vulnerability:
The most popular and straightforward way to replicate would be using burp suite. By intercepting the request and changing the value of the amount parameter to 0.01, you will theoretically be able to acquire as many tokens as you would like.

Note: Due to ethics and operational security reasons, I did not actively exploit this vulnerability , but can confirm the vulnerability exists here:
https://cdn.discordapp.com/attachments/794230066871009302/796109920893534218/Kazam_screencast_00001.gif

PHPMyAdmin and PHPMyAdmin setup/index.php

CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:H/A:L CVSS Score: 4.3 CWE-ID: CWE-264

The directory http://api.unknownvpn.net/phpMyAdmin/setup/ was open without any restrictions. There was no authentication required, which means that an attacker could view and modify all databases on the server. This also gives the attacker the ability to delete and add custom databases within the PHPMyAdmin instance. This allows the attacker to fill the server with junk data to the point that the server cannot store any important logs/userinfo/new users/etc. This can lead to a Denial Of Service attack.

The attacker can also edit and add malicious code to the Table structures with privileges as shown above to create a Remote Code Execution vulnerability.

Impact:
Attackers gain access to databases and therefore the confidential userdata and logs.
Attackers will have Read/Write?modify permissions on all databases hosted on the server.
Attackers can flood the server with random data by adding their own databases which results in lack of space and denying new log or userdata. (DoS)

Solution:
Limiting the request allowed to the /PHPMyAdmin/ directory to a certain IP and by requiring authentication for the setup page and forbidding the /phpmyadmin/setup.php file and with it the PHPMyAdmin directory as a whole.

Sensitive Information Disclosure

CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N CVSS Score: 5.3 CWE-ID: CWE-200

Within the source code of the UnknownVPN Client, which is open sourced and available at https://github.com/Unknown-Industries-LLC/UnknownVPN-Client, there is disclosure of information that should be kept private.

Firstly, the IP Address of every VPN server is hardcoded into the application. This can be used to form a blacklist, preventing those IPs from making connections to a service.

The backend IP Address to the API server is also hardcoded into the software, despite the presence of it under the cloudflare protected subdomain http://api.unknownvpn.net/. This opens the door for enumeration of services, directories, files and more with no limit to requests.

```
UnknownAPI.cs:        public static Dictionary<string, ServerObject> Servers = new
Dictionary<string, ServerObject>();
UnknownAPI.cs:            Servers.Add("[Game] OVH - Que CA 1", new ServerObject {
SoftetherConnectionName = "OVH1", IP = "167.114.58.193", RegionCode = "CA" });
UnknownAPI.cs:            Servers.Add("[Game] OVH - Que CA 2", new ServerObject {
SoftetherConnectionName = "OVH2", IP = "158.69.11.135", RegionCode = "CA" });
UnknownAPI.cs:            Servers.Add("[Game] OVH - Que CA 3", new ServerObject {
SoftetherConnectionName = "OVH3", IP = "167.114.29.161", RegionCode = "CA" });
UnknownAPI.cs:            Servers.Add("[Game] OVH - Lon GB 1", new ServerObject {
SoftetherConnectionName = "OVH4", IP = "51.89.176.238", RegionCode = "GB"});
UnknownAPI.cs:            Servers.Add("[Streaming] CHPA - Los US West 1", new
ServerObject { SoftetherConnectionName = "CHPA1", IP = "149.28.85.22", RegionCode =
"US" });
UnknownAPI.cs:            Servers.Add("[Streaming] CHPA - Flo US East 1", new
ServerObject { SoftetherConnectionName = "CHPA2", IP = "137.220.56.195", RegionCode
= "US" });
UnknownAPI.cs:            Servers.Add("[Streaming] IPC - AFR SY East 1", new
ServerObject { SoftetherConnectionName = "IPC1", IP = "45.141.58.93", RegionCode =
"AF" });
UnknownAPI.cs:        public static string ServerIP = "167.114.215.80";
```

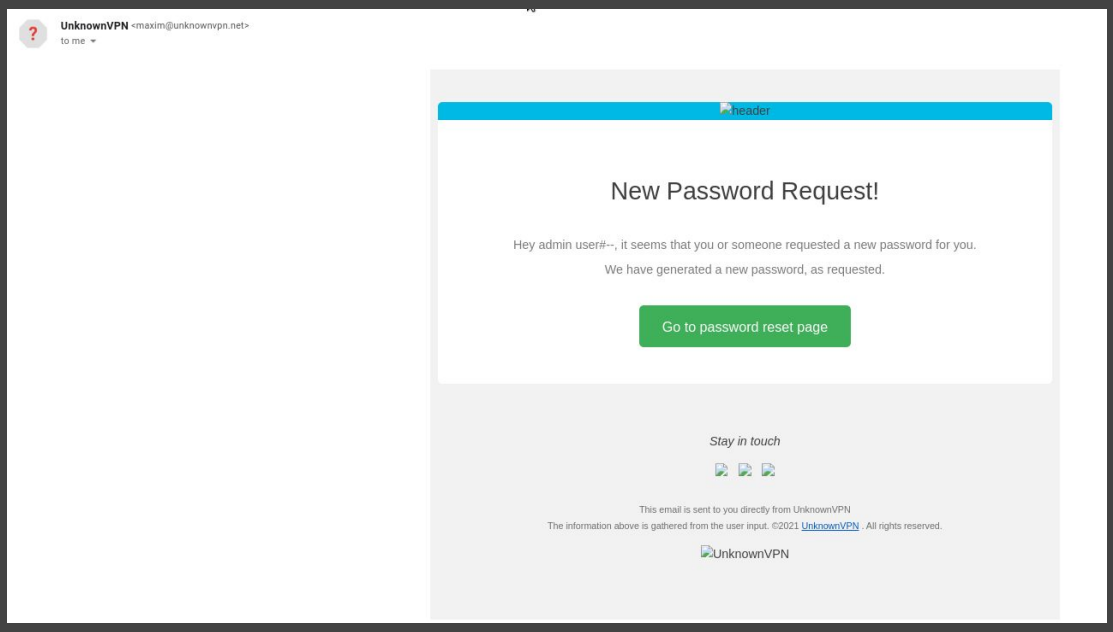
Solution:
Have the service obfuscate the identity of it's VPN services and hardcode the ServerIP variable towards the api subdomain.

High Impact Vulnerabilities

This section will describe the High Impact Vulnerabilities found within UnknownVPN and UnknownVPN related software. By definition, a High Impact Vulnerability is one with a calculated CVSS score of 7.0-8.9. The CVSS vector string will be provided with all vulnerabilities.

Weak Password Recovery Mechanism for Forgotten Password

CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:L CVSS Score: 7.3 CWE-ID: CWE-640



The task of resetting your password is poorly made. You just get an email with a unique directory and you can change your password in that directory.
The problem is that the site does not require any cookies or any other sort of authentication. So by resetting someones password and running a directory fuzzing or searching tool on http://unknownvpn.net/password/********/
The found page will let us change the password with no questions asked and grant us total account takeover.

Impact:
Account takeover with password resetting. The attacker can also change the email when he gets into the account. This results in account takeover and leaving the victim locked out indefinitely with no way of resetting the password.

Solution:
Implement a token and cookie system and a time based system on the password recovery so that anyone with that link cannot access someone else's password recovery page.

File Upload of a Dangerous File (requires bypass)

CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:L CVSS Score: 7.3 CWE-ID: CWE-434

Within the dashboard, there is a form that allows you to upload an avatar. This can be leveraged to upload a potentially dangerous file and execute code on the system.

There are checks on the file to prevent you from either:

A. Directly uploading a php.
B. Changing the mime type of the php.
C. Using nullbyte characters.

However, you can edit the contents of the file and use the utility exiftool to place your executable code within an existing image and upload it successfully.

To replicate you would perform the following:

```
cp /usr/share/images/kali-logos/logo-128.png shell.png
exiftool -Comment "<?php echo '<pre>'; system($_GET['cmd']); ?>"  shell.png
mv shell.png shell.php.png
```

After performing those steps, you will be able to upload the file as your avatar. After uploading, you will refresh the page, right click on the avatar, inspect element and get the address of the photo. From there, you paste it in the search bar, append your ?cmd=, and begin executing commands on the remote host.

The impact would be much greater if there was room to escalate, but given the site is hosted with cpanel, rather than being its own host, there is no potential for escalation and the impact is low.

Solution:
Just remove the functionality of avatars as a whole. It is a redundant feature that serves no purpose on the website and only presents a security risk.

Critical Vulnerabilities

This section will describe the Critical Vulnerabilities found within UnknownVPN and UnknownVPN related software. By definition, a Critical Vulnerability is one with a calculated CVSS score of 9.0-10.0. The CVSS vector string will be provided with all vulnerabilities.

Complete Account Takeover

CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:H/I:H/A:H CVSS Score: 9.6 CWE-ID: CWE-284

When inside the dashboard, you will notice that you have the ability to change your email address, password, and associated name all at once. With this single form, you have the ability to complete a complete account takeover on the main website that will merge all information between the 2 accounts.

If User1 performs the account takeover on User2 and User2 has a key for the UnknownVPN client, this will enable User1 to steal the key and obtain the service for free at the expense of User2.

We were unable to test if this account takeover is effective without burp, but can 100% confirm it's exploitability within it. To replicate, set a password within the form and submit while running Burp Proxy. When the request is intercepted, you will change the value in the email variable to the victim's email (Note: you must URL encode the @ symbol; it translates to %40) and record the password that you used. After submitting the request, you will find that the details of both accounts have merged and you have successfully performed the exploit.

Solution:
One main issue throughout the entire site is the lack of validation for actions done by the end user. There is no email verification for password resets which lead to an earlier vulnerability. This coupled with the ability to change the email within the same form allow you to submit the request as another user with absolutely NO restrictions. This was a common trend in previous vulnerabilities and will also show up in the next section.

General Security Issues

These are general issues with the security of UnknownVPN or any UnknownVPN related services. These could include those with a CVSS Score of 0.0, those that cannot be further tested, or issues with the service that need to be addressed.

Logging Policy

CWE-ID: CWE-319, CWE-532

Their logging policy does not fall in form with what they advertise as both the API Server and the Client application both fingerprint important and personally identifiable data.

Within the Client application, static code analysis reveals the presence of a hardware ID that contains most of your PC's information. On linux systems, you can see the scope of most of the logged data by running `grep -r "ID"` within the API folder in the repository. My output to this command is as follows:

```
└──$ grep -r "ID"
UnknownAPI.cs:                fingerPrint = GetHash("CPU >> " + cpuId() +
"\nBIOS >> " + biosId() + "\nBASE >> " + baseId()); //+"\nDISK >> "+
diskId() + "\nVIDEO >> " + videoId() + "\nMAC >> " + macId());
UnknownAPI.cs:        #region Original Device ID Getting Code
UnknownAPI.cs:            if (retVal == "") //If no UniqueID, use
ProcessorID
UnknownAPI.cs:        //Main physical hard drive ID
UnknownAPI.cs:        //Motherboard ID
UnknownAPI.cs:        //Primary video controller ID
UnknownAPI.cs:        //First enabled network card ID
UnknownAPI.cs:        public static Tuple<bool, string> Login(string user,
string pass, string uID)
UnknownAPI.cs:            var url =
$"http://{ServerIP}/auth2.php?user={user}&pass={pass}&uid={uID}";
UnknownAPI.cs:        public static bool Registration(string user, string
pass, string email, string uID, string token)
UnknownAPI.cs:            var url =
$"http://{ServerIP}/adduser2.php?user={user}&pass={pass}&email={email}&uid=
{uID}&token={token}";
```

The collection of such data violates the policy that the company claims to have.

Another violation of the logging policy is the presence of Apache2 logs. By reading access.log, anyone with permissions could find the time they logged in, the ip they logged in from and the credentials used during the attempt. This can be mitigated by using post to send data to the web server rather than putting plaintext creds within the get request.

Weak Captcha

CWE-ID: CWE-804

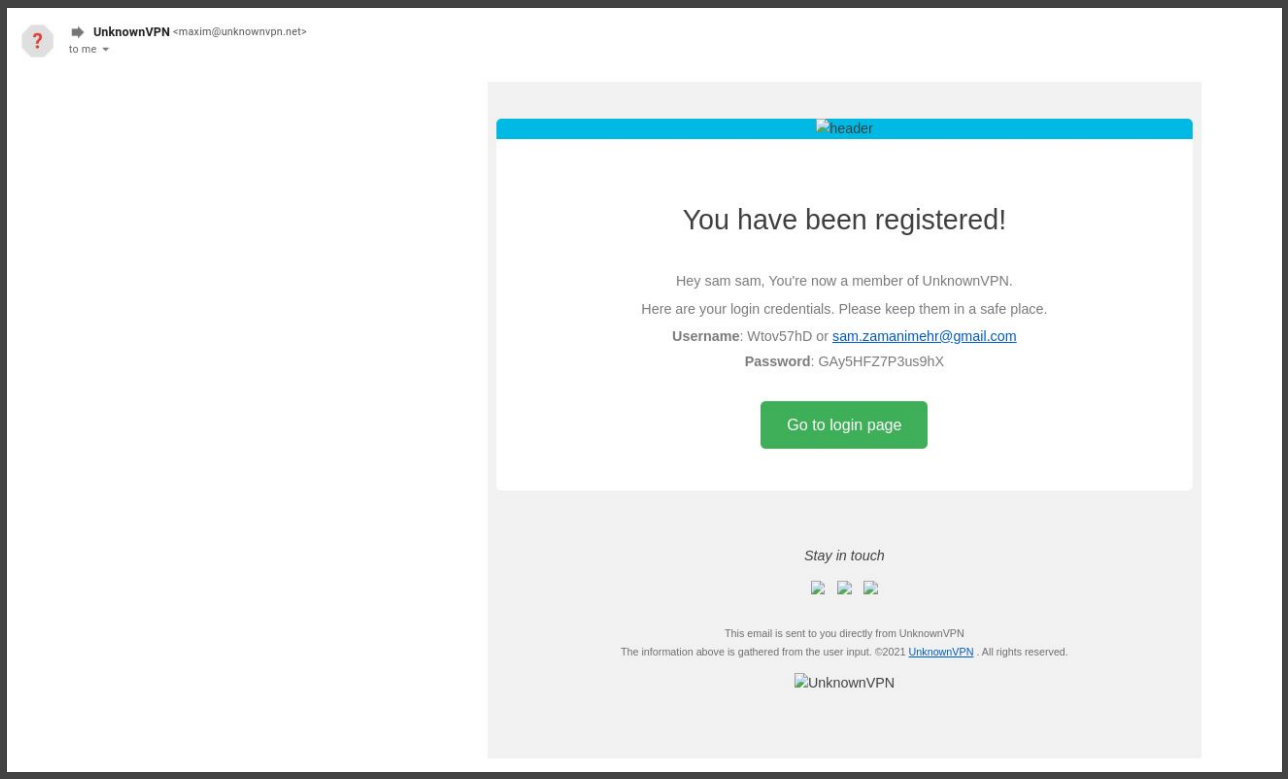
In every form associated with the website, the captcha appears to be javascript generated (cannot verify, we never parsed sources) and falls under the guessable captcha vulnerability class.. If you were able to obtain the backend IP Address, you would be able to perform flooding on the login page and the reviews page. This could impact availability of the service if you are able to use significant bandwidth, causing DoS.

A solution is to implement a known Captcha solution, such as google's. reCAPTCHA. It is easy to set up and deploy, alleviating any possibility of mass account or comment creation.

Insecure Password Storage

CWE-ID: CWE-522

We realized during the early part of the penetration test that the service had visual access to our passwords when we made a test account and we received an email containing our credentials in plaintext.



From there we could assume that our logins were being stored in something like base64, which has wide support and is easily reversible. After going further in the audit this was determined to be true.

This means that every user credential being used on anything within the scope of UnknownVPN can be reverse, which leads to 2 main issues: There is a breach of privacy between the service and the user, which violates the entire purpose of a VPN given user information is in no way secure, as well as the possibility of user credentials being made public or misused by an attacker.

For ethical reasons, we will not display the passwords in either base64 or plaintext, but we can verify their presence due to our possession of the user database.

A solution to this issue would be to use a hashing algorithm, as passwords should only be confirmed one way and the service has no right to the possession of the end user's plaintext credentials. To keep it secure and protect the end user, the hashing needs to be done on client side before it is sent to the server in whatever hashing algorithm was used, preventing the service from ever knowing your credentials.