Investigation into the security and privacy of iOS VPN applications

Jack Wilson
David McLuskie
Ethan Bayne

https://doi.org/10.1145/3407023.3407029

Investigation into the Security and Privacy of iOS VPN Applications

Jack Wilson
Division of Cybersecurity
Abertay University
Dundee, UK
hi@jack.lu

David McLuskie
Division of Cybersecurity
Abertay University
Dundee, UK
d.mcluskie@abertay.ac.uk

Ethan Bayne
Division of Cybersecurity
Abertay University
Dundee, UK
e.bayne@abertay.ac.uk

ABSTRACT

Due to the increasing number of recommendations for people to use Virtual Private Networks (VPNs) to protect their privacy, more application developers are creating VPN applications and publishing them on the Apple App Store and Google Play Store. In this 'gold rush', applications are being developed quickly and, in turn, not being developed with security in mind.

This paper investigated a selection of VPN applications available on the Apple App Store (for iOS devices) and tested the applications for security and privacy issues. This includes testing for any traffic being transmitted over plain HTTP, DNS leakage and transmission of personally identifiable information (such as phone number, International Mobile Equipment Identity (IMEI), email address, MAC address) and evaluating the security of the tunneling protocol used by the VPN.

The testing methodology involved installing VPN applications on a test device, simulating network traffic for a pre-defined period of time and capturing the traffic. This allows for all traffic to be analysed to check for anything being sent without encryption. Other issues that often cause de-anonymization with VPN applications such as DNS leakage were also considered.

The research found several common security issues with VPN applications tested, with a large majority of applications still using HTTP and not HTTPS for transmitting certain data. A large majority of the VPN applications failed to route additional user data (such as DNS queries) through the VPN tunnel. Furthermore, just fifteen of the tested applications were found to have correctly implemented the best-recommended tunneling protocol for user security.

Outside of the regular testing criteria, other security anomalies were observed with specific applications, which included outdated servers with known vulnerabilities, applications giving themselves the ability to perform HTTPS interception and questionable privacy policies.

From the documented vulnerabilities, this research proposes a set of recommendations for developers to consider when developing VPN applications.

CCS CONCEPTS

• **Computer systems organization → Security and Privacy**; Network Security; Security Protocols;

KEYWORDS

Mobile, Security, Privacy, Virtual Private Network, VPN, iOS

ACM Reference format:

Jack Wilson, David McLuskie and Ethan Bayne. 2020. Investigation into the Security and Privacy of iOS VPN Applications. In *Proceedings of the 15th International Conference on Availability, Reliability and Security (ARES 2020), August 25-28, 2020, Virtual Event, Ireland. ACM, New York, NY, USA, 9 pages.* https://doi.org/10.1145/3407023.3407029

1 INTRODUCTION

Virtual Private Networks (VPN) were originally designed for businesses to fulfil two primary purposes. Firstly, to allow an employee encrypted remote access to internal corporate networks and services. Secondly, to bridge the network of geographically separated corporate sites together over the internet (site to site VPN).

ARES 2020, August 25–28, 2020, Virtual Event, Ireland

ACM ISBN 978-1-4503-8833-7/20/08…$15.00
https://doi.org/10.1145/3407023.3407029

Recently, companies have begun offering VPNs as a service, where users can route all internet traffic through the VPN provider's servers. Routing all traffic through an encrypted VPN tunnel to the VPN server minimizes risk of data interception from Internet Service Providers (ISPs), while offering better privacy and/or security when communicating on open or untrusted networks.

VPNs have been gaining popularity over recent years [1]. This growing popularity could be partially attributed to media outlets recommending readers to adopt VPNs to avoid tracking by governments and ISPs. There are many examples of similar recommendations and articles both online and in print [2]. While there is accuracy to some claims in the various articles on the subject, articles tend to offer poor recommendations on what features constitute a secure VPN product.

A VPN product may not offer a true anonymous service due to security weaknesses within their VPN client and/or privacy policies. Common examples where a VPN provider may de-anonymize a VPN user may be through logging IP addresses for troubleshooting, recording analytics data, or from requiring an email address and/or payment details for a VPN account.

This research aims to evaluate VPN applications available on Apple iPhone's iOS mobile operating system to establish the state of security and privacy offered by VPN providers. To establish this, a methodology was built to assess the security and privacy of VPN applications and VPN privacy policies. The applications that were selected for this research were either free to use or offered a limited trial period. All applications were obtained on iOS 11 from the Apple App Store. The authors of this paper are unaware of similar research that investigates the privacy of VPN clients on the iOS platform.

The contribution of this paper is threefold:

- To provide an overview of the state of security within free and paid for iOS VPN clients.
- A presented methodology that can be used to evaluate the state of the security of VPN clients.
- A set of guidelines of recommended practices are outlined that will allow iOS developers to create secure VPN clients.

2 BACKGROUND

At present, businesses are becoming increasingly data driven, often requiring consumers to surrender personal information to use their services. With user data being held by more businesses, this has made privacy a more prevalent issue. Studies on user privacy and mobile application privacy have been undertaken which indicate users are becoming less concerned about their privacy while mobile applications are sending more user data than ever.

Research from Ridley-Siegert [3] asked a selection of people their opinions on how their data was used and handled. The study found that participants of the study were less concerned about their online privacy to a previous 2012 study, showing a reduction of the overall concern from 84% to 79% between 2012 and 2015.

Alongside user attitudes towards privacy, developer attitudes towards how users and their data are treated by developers must also be considered (this is an influencing factor to the research being undertaken). Developers should ensure they are not taking an excessive amount of unnecessary data and must also ensure that the data is being handled and stored securely.

Prior research analyzed different versions of Android applications, with releases spanning from 2011 to 2015 [4]. The research investigated what user data and device information was being sent to the developers and third parties. The results from this study demonstrated that, overall, applications have increased how much user data is being sent to developers.

2.1 Definition of Personally Identifiable Information

Personally Identifiable Information (PII) is a term that is used to describe any form of information that could identify an individual person or device. Ensuring that PII does not end up in the hands of an adversary is of critical importance. If, for example, a user was using a VPN for security and privacy on public WiFi but the VPN service was transmitting a username and password over HTTP, an attacker sniffing traffic on the network may be able to steal the credentials and then impersonate the user.

The Information Commissioner's Office (ICO) have created a flowchart-style document that can quickly determine if an item of information could be deemed 'Personal Data' [5]. This document was partially referenced to create the list of PII as shown in Figure 1.

- Device Identifier
  - IMEI
  - IP Address
  - Device Serial Number
- User Identifier
  - Name
  - Banking Details
  - Date of Birth
  - Contact Information (Phone Number, Email Address etc.)
- Location Data
  - Home/Work Address.
  - GPS Location
- Credentials
  - Username
  - Email Address
  - Password

Figure 1 PII Data

2.2 Prior Research

Prior research into VPN security has been undertaken for mobile and desktop clients at a variety of price points. One such paper is An Analysis of the Privacy and Security Risks of Android VPN Permission-enabled Applications [4], which investigates several common issues (which will be described in the next section) that can reduce the privacy of VPN users on the Android platform. The paper covered a large dataset of 283 applications, but only focused on VPN clients within the Android operating system.

A Glance through the VPN Looking Glass [6] is another piece of VPN security research that investigated a significantly smaller dataset (just 14 VPN clients) but clients were analyzed on every common operating system (iOS, Android, Windows, OS X and

Linux). The research involves testing the selection of VPN clients for known security issues within VPN clients (such as IPv6 leakage and DNS leakage).

The aforementioned papers show that there are issues present within the VPN client ecosystem. The research outlines a variety of the vulnerabilities present that could reduce the privacy of a user and discuss methods to test for privacy vulnerabilities.

2.3 VPN Privacy Issues

There are several common security issues with VPN clients that, if present, could reduce the privacy of a user. This section does not consider security vulnerabilities of individual VPN client applications, but rather on common security issues through misconfiguration errors or poor software design.

2.3.1 DNS Leakage

A DNS leak is the result of a VPN client making DNS requests to a DNS server not controlled by the VPN provider. Making DNS requests to a third-party – most commonly an ISP's DNS server – allows that third-party to view the websites or services a user is visiting based on the hostnames that the DNS server is being asked to resolve.

If a VPN is being used to avoid tracking by ISPs, then a DNS leak to an ISP's DNS server could negate the purpose of the VPN service. The reason for this is that an ISP can monitor and record the DNS resolve requests being made from a user's source IP address (outside of the VPN tunnel) [7].

If a VPN provider was to run their own DNS server, then this would mitigate the issue of a DNS leak as it centralizes trust with the developer rather than trusting a third-party DNS provider.

2.3.2 Tunneling Protocols

iOS supports a selection of VPN tunneling protocols. Each protocol has a variety of benefits and disadvantages, some are more secure than others and each has support for different authentication mechanisms [8]. The purpose of investigating this security issue is to determine if the VPN clients are using a secure, modern tunneling protocol (such as a variant that utilizes IPSec) or a less secure, older protocol such as L2TP.

2.3.2.1 IPSec

IPSec is a protocol that is often combined with other protocols (detailed below) that provides enhanced encryption and better authentication mechanisms. IPSec relies on two main protocols. The Authentication Header (AH), which is used in IPSec to verify the source of the traffic and utilizing hashes of the sent data to ensure that traffic is not tampered with in transit. The second protocol is the ESP which uses symmetric encryption to encrypt data being sent between clients [9].

IPSec has three methods that can be used for authentication: pre-shared keys, encrypted nonce or digital certificates. These authentication mechanisms are detailed in the authentication mechanisms section.

2.3.2.2 IKEv2 (with IPSec)

IPSec is responsible for the integrity checking and encryption part of a VPN tunnel, but this is often combined with Internet Key Exchange (IKEv2) to exchange keys for authentication purposes. This mutual authentication is performed by establishing a security association between two parties, in this case the user and the VPN server, where the shared secret information is exchanged to allow for the communication of the aforementioned AH and ESP protocols [10].

2.3.2.3 L2TP (with IPSec)

The Layer 2 Tunneling Protocol (L2TP) is a slightly older tunneling protocol that is an advancement of the Point-to-Point Tunneling Protocol. It was built in a partnership between the PPTP Forum, Cisco and Internet Engineering Task Force (IETF). The L2TP protocol is defined in RFC2661 [11]. By default, L2TP does not encrypt any traffic, as it is used mainly to establish the VPN tunnel. For this reason, L2TP is often combined with IPSec and is known as 'L2TP over IPSec'. Securing the L2TP protocol using the IPSec suite is defined under RFC3193 [12].

2.3.2.4 SSL VPN

Secure Socket Layer (SSL) VPNs are primarily intended for employees to remotely access internal resources within a company. An SSL-based VPN typically uses port 443 (HTTPS), so there are often no issues with firewalls as this is a well-known port number. SSL VPNs are browser-based so there is no need for any operating system integration or third-party VPN clients. This also makes them very lightweight in terms of CPU/RAM utilization when compared to other VPN tunneling protocols [13].

2.3.3 Authentication Mechanisms

Authentication is a key part of a secure VPN infrastructure that ensures only valid and legitimate users of a VPN service can access the service. There are three common authentication methods which are detailed in the sections below

2.3.3.1 Pre-Shared Keys

A pre-shared key (PSK) is essentially a password and it authenticates in a similar fashion that a user would connect to a basic home WiFi network, where one shared 'password' is used for every user that connects.

If PSK's are used for authentication and the PSK is not unique for every user, this creates a security weakness. An adversary/attacker could impersonate the VPN server and route all traffic through the adversary's VPN server [14]. It is therefore recommended not to use non-unique pre-shared keys for authentication.

2.3.3.2 Encrypted Nonce

A nonce is constructed from randomly generated and unique numbers and mitigates the concerns of utilizing non-unique PSKs. In the context of VPN authentication, the VPN provider and the user must exchange a key to encrypt user's nonce with the VPN provider's public RSA key. Given that the nonce are unique this can help to mitigate replay attacks [15].

2.3.3.3 Digital Certificates

Digital certificates work in a similar way to an encrypted nonce, relying on asymmetric cryptography to encrypt and decrypt data [16]. The difference is that digital certificates are issued by a trusted certificate authority (in the same way SSL certificates are issued). The main advantage to issuing certificates in this way is that distribution of certificates is centralized with trusted certificate authorities. This also allows certificate authorities to revoke certificates that may have been compromised [17].

2.3.4 IPv6 Leakage

Another common issue present within VPN clients that could reduce the privacy of a user and allow them to be tracked is through IPv6 leakage. This issue occurs when the operating system makes requests using IPv6 rather than IPv4. Due to a lack of support by some VPN clients, IPv6 requests are routed outside of the secure VPN tunnel, potentially leaking user information. This issue is also detailed in A Glance through the VPN Looking Glass [6].

On the iOS platform this issue should theoretically be relatively minor, if a tunneling protocol is in use that does not support IPv6 then IPv6 is disabled system wide. The only VPN tunneling protocol that supports IPv6 is IKEv2 [8].

2.4 VPN Anonymity

Using a VPN can offer a user more privacy, assuming that the VPN service is secure and does not suffer from the issues discussed in the previous section. However, some VPNs do not make a user untraceable, and users that employ VPNs for communications may be far from being anonymous [18].

Most VPN providers require an account to use the service, typically asking for a name, email address or payment information on account creation. A user could be personally identified from the account information provided. Even VPN providers that do not require accounts will often log users' source IP addresses, which could also be used to deanonymize a user.

WebUser magazine recently released an issue with the front page boldly claiming VPNs would allow users' to 'stay 100% anonymous' while offering guidance on what a VPN is and why consumers should use a VPN service [19]. The general explanation was technically accurate, and a variety of VPN services were recommended at different price points. However, the 'stay 100% anonymous' claim is entirely false, due to the fact most VPN services will log PII, thus potentially removing any form of anonymity.

2.5 Threat Modelling

The threat model of a VPN user must be heavily considered before using and purchasing a VPN. Security conscious users, political activists, and other users wishing to circumvent geographic restrictions to access content may use a VPN. However, as acknowledged previously, VPNs are far from anonymous due to the information requested by the VPN provider to use the service.

If the user's objective is to avoid nation state actors or law enforcement, the majority of VPN services will not protect the user from this. Many VPN providers have terms in their privacy policies that claim they will cooperate with law enforcement investigations. One such example is VyprVPN (a product owned by Golden Frog), which states in its privacy policy (under the *How Golden Frog Responds to Criminal Investigations* section):

"Golden Frog cooperates fully with law enforcement agencies, yet there must still be a subpoena before Golden Frog provides a member's identifying information - minimal information reasonably calculated to identify and no more" [20].

3 Procedure

This procedure outlines the approach to the case study performed as part of this research. The procedure tested for prevalent technical issues that can affect the privacy of VPN applications. Each stage of testing performed is presented to ensure that the test criteria is met and to ensure reproducibility.

The applications tested throughout the research were downloaded from the Apple App Store. The applications were advertised to be either completely free to use or offer a free trial. The applications were found in the Apple App Store by using the search queries; "Free VPN", "Free VPN for iPhone", "Secure VPN", "Free VPN Anonymous", and "Fast Free VPN".

3.1 Device Preparation

To test the applications, an iPhone 6 was used as a test device that was running iOS 11.2.2. Automatic updates for both the operating systems and installed applications were disabled to prevent any possible changes to the results throughout the testing period.

During the testing of each VPN application, the only other application open was the Safari web browser. The Safari web browser was used to test for DNS leakage using an online DNS leak test tool. No other tabs were open in the browser and no other applications were running in the background to mitigate collecting unnecessary data that could skew collected results.

The iPhone was connected to WiFi and had no SIM card inserted. This ensured that all data was sent over WiFi, and no data was transmitted using a mobile data connection. Internet traffic was also mirrored over USB to a computer to capture traffic.

3.2 Accounts within Applications

It was determined that there were three categories of accounts within VPN applications:

1. An account was required.
2. An account was optional.
3. No option to have an account.

Accounts were created in cases where an account was required or optional to use the VPN service. This was done to determine if any sensitive PII were being transmitted over unsecure HTTP. Naturally, when no option for an account was required then no account was created.

3.3 Test parameters

All VPN applications were tested for the following areas of security and privacy:

- Data communication (HTTP/HTTPS).
- DNS leakage.
- Secure tunneling protocols.
- Application permissions.

3.4 Packet Capture

The study used a technology called Remote Virtual Interfaces (RVI) to facilitate packet capture from the iOS device. This technology exists exclusively on iOS 5 or later and requires a device running OS X/MacOS to use for packet capturing. The iOS device had to be connected to the MacOS device using a lightning to USB cable.

With the prerequisites installed (Xcode), running the command (shown in Figure 5) in a terminal window created a virtual network interface that mirrored the iOS network traffic to a virtual network interface on the MacOS device.

```
rvictl -s <device UDID>
```

Figure 2: Command to Start Remote Virtual Interface

The Unique Device Identifier (UDID) was determined by connecting the test device to iTunes and retrieving it from the phone information section. The -s flag creates the network interface and the -x flag with the same command stops and removes the virtual network interface. Once the command was executed, packet capturing software (such as Wireshark) could be directed at the virtual network interface to allow all traffic to be captured.

Compared to using a proxy or ARP spoofing, this was the most effective method to packet capture traffic as it made no difference which subnet each device was on. An additional benefit was there was no risk of interfering with other user's network traffic by packet capturing WiFi traffic as the iOS test device is connected to the laptop.

3.5 Analysis of PCAP Files

The analysis of the captured PCAP files involved several steps. The first step analyzed HTTP traffic to check if the application was using the HTTP protocol, and if so, whether the HTTP traffic contained any data that could be deemed personally identifiable. If HTTP was not in use, then HTTPS was used and the application was deemed secure from leaking PII in plaintext.

Due to the sheer amount of PCAP files accumulated over the course of the testing, it became infeasible to manually analyze each file for the presence of PII. For this reason, a small bash script was written to parse the PCAP files for a list of keywords that were accumulated through the initial manual PCAP analysis, including when applicable information used when setting up any accounts with the VPN application.

3.6 DNS Leakage

Testing for DNS leakage was handled through a third-party DNS leak test website—www.dnsleaktest.com. The VPN connection was tested once initiated and through an extended run test using the Safari web browser on the iPhone. The test performed made a series of DNS requests to determine what DNS provider was being used to serve the DNS requests. If any DNS requests were made to any servers other than those in control by the VPN provider, then the application was determined to be suffering from DNS leakage. The DNS cache was cleared between testing each application.

3.7 Tunneling Protocol

Ideally, VPN applications should be making use of the ESP protocol within the established VPN tunnel. To establish whether this was present, a network security monitor called Bro, was used to analyze each PCAP file after the testing was completed. This was done to offer further analysis and a breakdown of the protocols found in the PCAP file.

3.8 Application Permissions

The OS permission requests being made by the VPN applications were analyzed by interacting with the application and recording which OS permissions the application requested. Some permission requests (such as VPN and notification) are expected, however, this section of the testing procedure was to determine if any applications were requesting permission they did not necessarily require (such as access to the media library or to contacts on the device).

4 Results

This section details the overall results of testing 57 VPN applications, outlining how the applications were categorized in the Apple App store, what permissions the VPN applications were requesting, DNS leakage, tunneling protocols that are used as well as examples of PII leakage. The results will additionally review some of the more interesting observations discovered while undertaking testing.

4.1 Overall Results

A total of 57 VPN applications were chosen based upon their popularity in the Apple App store, ranging from 1* to 5*. It was observed that 13 of the least popular VPN applications available on the Apple App Store were generally less reliable than popular applications, frequently failing to establish a connection to the VPN service.

4.1.1 VPN Categories

The Apple App Store does not have one definitive category for VPN clients, so it was interesting to see which categories application developers chose to categorize their applications under. Figure 3 shows the results of how the applications were categorized. The productivity and utilities categories on the Apple App Store were found to be the most popular choices for categorizing VPN applications, however, a few applications were additionally found under travel, business, and reference categories. There were some applications which were under no category at all.

4.1.2 Permission

Through interacting with applications during testing, the permissions applications were requesting or requiring was recorded. 6 out of the 57 applications tested were not able to have their permissions recorded as they were unable to connect to their VPN server.

Every application naturally required the VPN permission, 17 of the applications were also requesting permissions to send notifications (e.g. if the application had a set amount of data per month, notifications were used to inform users when their data allowances were running low).

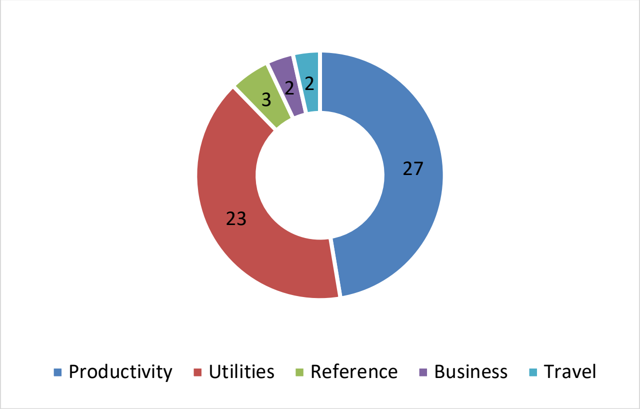

Figure 3: Breakdown of Application Categories

One application required access to GPS to determine user location to find the fastest servers to connect to. The only other permission requested by any application was by application #56, which required access to user's photo libraries. The permissions are broken down in Figure 4.

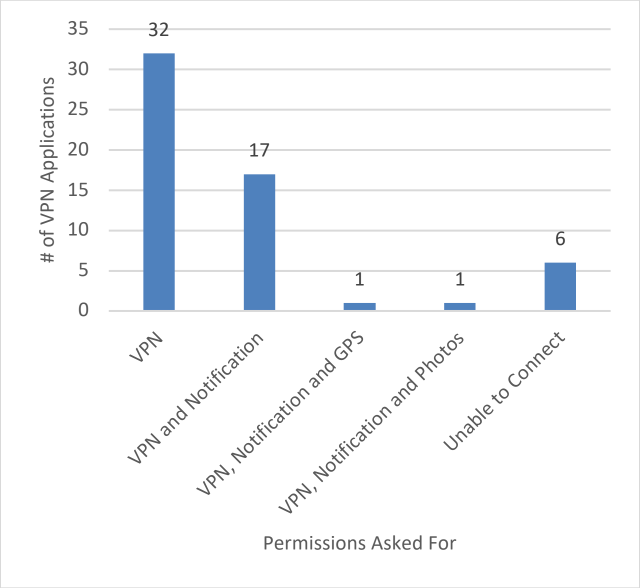

Figure 4: Application Permission Requests

4.1.3 HTTP within applications

Figure 5 shows that out of the 57 applications tested, 40 of these were determined to be using the HTTP protocol (regardless of whether the HTTP traffic contained PII or not).

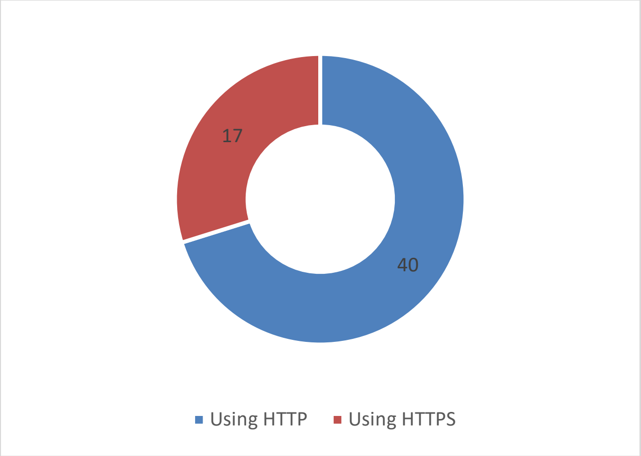

Figure 5: Applications using HTTP or HTTPs

4.1.4 PII Leakage

By using keywords within the bash script and verifying the script's results, Figure 6 shows that 22 of the 57 tested applications were leaking PII about users. This information included details such as usernames, email addresses, passwords, source IP addresses and GPS coordinates. The GPS coordinates were either determined from geolocation information deduced from the source IP address or from the reported device location obtained through GPS permissions.

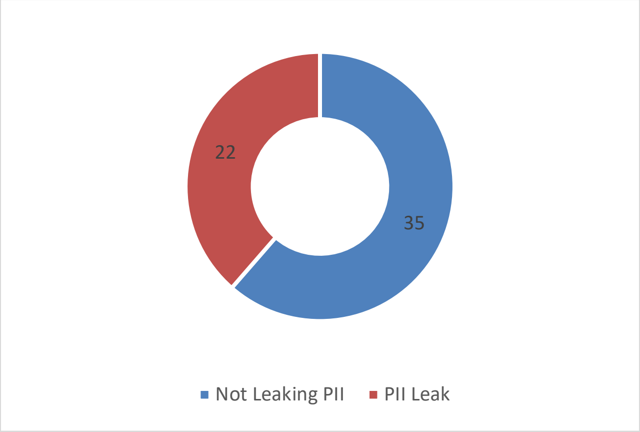

Figure 6: Chart of PII Leakage Results

4.1.5 DNS Leak within applications

Figure 7 shows that 13 of the 57 VPN applications tested were able to communicate with their VPN server, but ultimately failed to establish a VPN connection. As such, 44 applications were tested for DNS leakage. Of the 44 applications that were tested, 32 suffered from a DNS leak. 30 of the VPN applications that suffered from a DNS leak were using Google DNS and 2 VPN applications were relying on Level 3 Communications (a large telecommunications provider) for DNS. This indicated that the respective DNS provider that DNS was being leaked to could associate DNS requests with a user and, therefore, track websites that users visit.

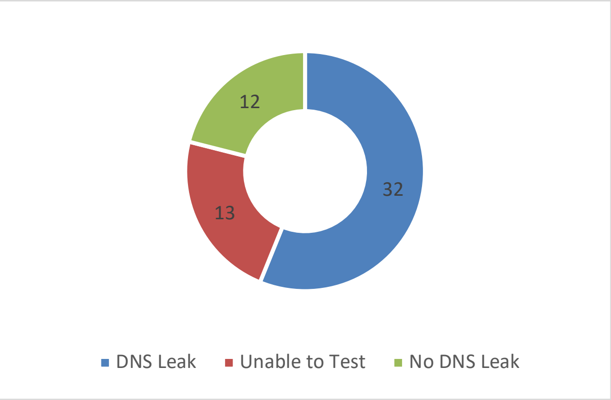

Figure 7: Chart of DNS Leak Test

4.1.6 Protocol Usage

Ideally, applications should be using the IPSec protocol with IKEv2 and ESP to ensure secure communications. The results from Figure 8 show that only 15 of the 57 VPN applications tested were using the recommended configuration of IPSec with IKEv2 and ESP. 17 applications were using IPSEC and IKEv2, 12 applications were using IPSEC and ESP and 1 application was just using IPSEC by itself. 5 applications were using another type of encryption and 7 of the applications could not be tested due to the fact that an VPN connection could not be established.

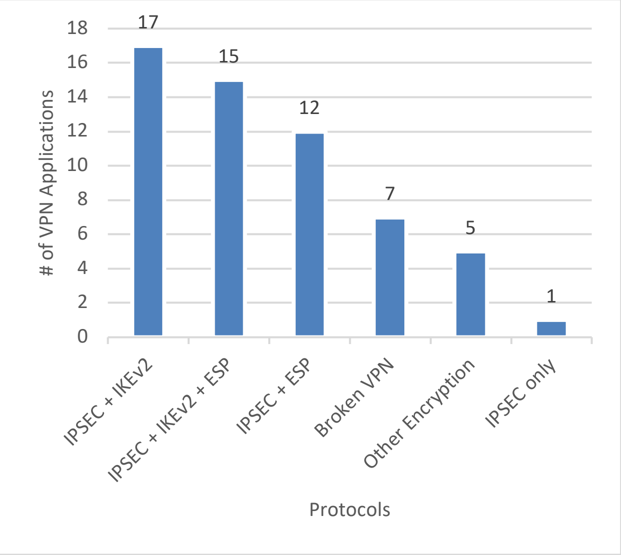

Figure 8: Chart of Encryption Protocols in Use

4.2 Examples of PII Leakage

This section outlines some of the more critical findings where applications were found to be leaking PII using the insecure HTTP protocol.

4.2.1 Application #5

During testing, application #5 was observed to be sending PII in URL parameters. This included the email and password used to create an account within the application. The 'pass' parameter was determined to be hashing the password using MD5 (this was verified by cracking the MD5 string and verifying it matched the original password). MD5 is a weak hashing algorithm[21] and, despite being a marginal improvement over transmitting passwords in plain text, it is still a insecure method of securely transmitting credentials.

Application #8 was observed to be sending PPI to a server, with the PII enclosed within XML files. This included an email address, a password and a unique device identifier.

4.2.2 Application #21

Application #21was observed to be sending a large quantity of PII over HTTP enclosed within a JSON file. This included the country, city, partial postcode and latitude and longitude coordinates of the user, as well as the user's IP address and ISP.

4.2.3 Application #52

Application #52 was determined to be transmitting sensitive information over HTTP, although not necessarily that of the user. The sensitive information transmitted contained an IP address (owned by a cloud hosting provider) and credentials used to authenticate the user to the VPN service.

4.3 Notable Observations

Some interesting results were discovered that were outside of the regular testing methodology. These results are outlined below and were reported to the developers as security issues.

4.3.1 Configuration file over HTTP

Application #52 was found to be downloading the VPN configuration file over HTTP. This method of configuration may expose the VPN application to man-in-the-middle attacks, where an attacker could intercept the configuration file and replace it with a malicious configuration file, thus routing all user VPN traffic through the malicious VPN server.

4.3.2 Self-Signed Root Certificate

During the setup process Application #17 requested permission to install a profile on the device. This profile was signed by 'VoiceFive Networks, Inc' with a verified tick.

Clicking on 'More Details' revealed that the profile contained VPN settings (as expected), four signing certificates that were issued by Symantec and Verisign (trusted certificate authorities on iOS), and two signing certificates that were issues by 'MobileXpression CA'. The list of trusted root certificates that are preinstalled on iOS does not contain MobileXpression CA, and this certificate was self-signed by the application developer.

Allowing a developer to install a self-signed certificate on the device is a security concern. This would theoretically allow the application developers to intercept encrypted HTTPS traffic and view confidential data such as usernames and passwords for other websites, applications and services. It may also allow the developers to inject adverts into webpages that users visit.

When continuing with the installation of the root certificate, iOS will specifically warn users that an unmanaged root certificate will not be trusted by default and that full trust will need to be enabled to proceed. To continue using the VPN application, enabling full trust is a requirement. No other application tested required that a self-signed root certificate was installed on the device, this was an anomaly and generally not the best practice for configuring a VPN client on iOS.

4.3.3 Outdated Web Server

When analyzing Wireshark packet captures, application #15 was determined to be making unusual requests to a directory on a web server. Upon visiting the directory using a web browser, it was discovered that the server was running Django, which is a Python web framework. The developer had left debugging enabled publicly, and the web page displayed debugging information (including the Django version number).

The specific version of Django was checked against the Common Vulnerabilities and Exposures (CVE) database for known vulnerabilities. It was determined that the installed version of Django was over 4 years old at the time of writing and had multiple vulnerabilities that had high severity ratings on the CVE rating scale.

5. DISCUSSION

This section discusses the results as well providing advice that iOS developers can take to improve the overall security of VPN applications based on the findings of this research. Implementation of the below advice would improve the privacy of a VPN client considerably over the VPN clients tested in this study.

5.1 VPN Categories

There are a large quantity of VPN applications in the Apple App Store. Within the tested results, there were inconsistencies with the categorization of applications. VPN applications were predominantly categorized under Productivity or Utilities, however, some applications within the selection that were tested were also categorized under Travel, Business and Reference.

This inconsistency in VPN application categorization could impact how applications are ranked, and what applications are displayed to users (e.g. when browsing specific categories for VPN applications). Due to the substantial amount of VPN applications available, it could be beneficial for Apple to introduce a category specifically for VPNs.

5.2 Permissions

The results of testing permissions were surprisingly uninteresting, with almost every application only requesting a combination of VPN and notifications permissions. There were only two applications that differed from this. Application #49 additionally required GPS permissions, which was used by the application to ‘find the closest Speed Servers’.

Although this does make sense from a technical standpoint, it stood out as the only application doing this through GPS location. Other applications (as evidenced through PII leakage) were able to determine approximate location through the user’s IP address rather than through explicit GPS coordinates from the device.

The second application observed to be requesting unusual permissions for a VPN was application #56. This application required additional access to the photo library. Upon further investigation, this permission was requested for another of the application’s intended functions that offers users to remove duplicate photos from their device. No further investigation was done to establish whether this application feature had malicious intention as this was beyond the scope of this research.

5.3 HTTP & PII Leakage within Applications

It is common for VPN applications to connect to multiple web servers on start up to authenticate VPN users, obtain server lists, and potentially receive news and ads. The research found that 40 out of the 57 VPN applications tested were using HTTP to facilitate this communication, which sends user information across the internet unencrypted.

Sending potentially confidential user information (such as usernames, passwords, phone IMEI numbers and advertising ID’s) across the internet without encryption serves to substantially reduce the privacy of a user.

Apple intended to enforce mandatory transport encryption for all iOS applications by the end of 2016 [22], but this decision was later postponed indefinitely. Even though this decision was postponed it is recommended that iOS developers should use mandatory transport encryption. When mandatory encryption is enforced it will greatly improve the overall level of security provided to the end user.

5.4 DNS Leakage

DNS leakage was another prevalent issue found over the course of application testing. Out of 44 VPN applications that were tested for this issue, 32 were found to be leaking DNS to a third-party organization. Given that a large majority of the DNS leaks were leaking requests to Google DNS servers, it introduces debate around user privacy, as the leaked DNS requests will enable Google to track the websites and services that the VPN user has visited.

The recommended best-practice to prevent a DNS leak is to ensure that all DNS traffic is routed through the VPN tunnel, and preferably to a secure DNS service that advertises an interest in user privacy and support for DNS over TLS. Some examples include the recently launched Quad9 DNS (built in part by IBM) and the Quad1 DNS resolver, built in collaboration with Cloudflare.

In terms of deployment from a developer perspective, setting the DNS resolver for a VPN tunnel involves implementing the NEDNSSettings class within the NetworkExtension framework.

5.5 Protocol Usage

Out of the 57 applications that were tested, 15 applications were found to be using the best-recommended practice of using the IPSec protocol with IKEv2 and ESP. Given the strong security of ESP and the features of IKEv2, it was surprising to see such a low number of applications supporting both protocols.

It is recommended that a VPN application should implement support for the IKEv2 protocol with the use of ESP, as this is the strongest cryptographically, while also supporting IPv6 and offering improved stability across network changes.

Deployment of this protocol relies on the use of the NEVPNProtocolIKEv2 and NEVPNProtocolIPSec classes within the NetworkExtension framework.

5.5.1 IPv6 Leakage

Following the recommendation to implement IKEv2 will result in the VPN having support for IPv6. Similar to DNS leakage, IPv6 leakage happens when IPv6 traffic is not properly routed through the VPN tunnel. It is recommended to add support for IPv6 through the VPN tunnel using the NEIPv6Route and NEIPv6Settings classes within the Network Extension framework.

5.6 Unnecessary Information Gathering

Throughout this research, numerous VPN applications were found to be leaking PII that could identify a VPN user. Combining this with the ever-increasing amount of data breaches affecting companies worldwide [23] , there is a substantial risk with handling and storing user information.

On the 25th of May 2018, the General Data Protection Regulation (GDPR) came into effect, which aims to ensure companies handle and store data securely to prevent data breaches. This is backed up by the threat of large financial penalties if companies were to suffer from a data breach.

To minimize the risk of data breaches it is important to ensure best practices are followed regarding server hardening and patching. Additionally, it is advisable for developers not to collect unnecessary user data and only gather user data required to provide the service.

6. CONCLUSIONS

In conclusion, this research has proven that some security flaws exist within a large selection of VPN applications on the iOS platform. One of the most notable problems found in the VPN applications tested was the lack of utilizing a secure communications protocol (HTTPS) to transmit data. This issue could be solved by Apple enforcing mandatory transport encryption for all iOS applications (which was announced and then delayed indefinitely), which would require all traffic to be sent over HTTPS to prevent sensitive information leakage [22].

Another prevalent issue found was DNS leakage, with 32 of 44 VPN applications tested suffering from DNS leaks. This issue leads to a substantial reduction in privacy for users, enabling the DNS providers to track users browsing activity based on DNS resolution requests. It is of critical importance to user privacy that VPN developers route all DNS requests through the VPN tunnel, while also employing a secure DNS provider.

It is crucial that developers understand the various risks and common security misconfigurations that can cause a reduction to user security and implement necessary fixes. The most common security misconfigurations have been presented and investigated throughout this paper. Recommended guidelines for mitigating privacy threats are presented to provide developers advice on improving privacy of their VPN applications.

REFERENCES

[1] Google. 2018. VPN. Retrieved 11 April, 2018 from https://trends.google.com/trends/explore?date=2010-11-04%202018-04-11&q=VPN.
[2] Eddy, M. 2019. What Is a VPN, and Why You Need One. PCMag. Retrieved from https://uk.pcmag.com/features/88655/what-is-a-vpn-and-why-you-need-one
[3] Ridley-Siegert, T. Data privacy: What the consumer really thinks. *Journal of Direct, Data and Digital Marketing Practice*, 17, 1 (2015), 30-35.
[4] Ikram, M., Vallina-Rodriguez, N., Seneviratne, S., Kaafar, M. A. and Paxson, V. *An analysis of the privacy and security risks of android vpn permission-enabled apps*. City, 2016.
[5] ICO What Is Personal Data?–a Quick Reference Guide (2012).
[6] Perta, V. C., Barbera, M. V., Tyson, G., Haddadi, H. and Mei, A. A glance through the VPN looking glass: IPv6 leakage and DNS hijacking in commercial VPN clients. *Proceedings on Privacy Enhancing Technologies*, 2015, 1 (2015), 77-91.
[7] dnsleaktest. 2018. What is a DNS leak and why should I care? Retrieved 3 March, 2018 from https://www.dnsleaktest.com/what-is-a-dns-leak.html.
[8] Apple. 2018. Intro to VPN with Apple devices. Retrieved February 28, 2018 from https://support.apple.com/en-gb/guide/deployment-reference-ios/ior9f7b5ff26/web.
[9] Barker, E., Dang, Q., Frankel, S., Scarfone, K. and Wouters, P. *Guide to IPsec VPNs*. National Institute of Standards and Technology, 2019.
[10] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P. and Kivinen, T. *Internet key exchange protocol version 2 (IKEv2)*. RFC 5996, September, 2010.
[11] Townsley, W., Valencia, A., Rubens, A., Pall, G., Zorn, G. and Palter, B. RFC2661: Layer Two Tunneling Protocol" L2TP" (1999).
[12] Patel, B., Aboba, B., Dixon, W., Zorn, G. and Booth, S. RFC3193: Securing L2TP using IPsec (2001).
[13] Hoffman, P. SSL VPNs: An IETF Perspective (2008).
[14] Bui, T., Rao, S., Antikainen, M. and Aura, T. *Client-Side Vulnerabilities in Commercial VPNs*. Springer, City, 2019.
[15] Li, J. Design of authentication protocols preventing replay attacks. *2009 International Conference on Future BioMedical Information Engineering (FBIE)* (2009), 362-365.
[16] McLuskie, D. and Belleken, X. 2018. X. 509 certificate error testing. In Proceedings of Proceedings of the 13th International Conference on Availability, Reliability and Security. Hamburg, Germany, 1-8.
[17] Rajakumar, J. and Subrahmanya, K. Overview of TLS Certificate Revocation Mechanisms. *International Journal of Advanced Research in Computer Science*, 10, 3 (2019).
[18] Dordal, P. L. The Dark Web. *Cyber Criminology* (2018), 95-117.
[19] Irvine, R. 2018. Stay 100% Anonymous VPNs The Ultimate Guide. WebUser, 443, 40-46. Retrieved from
[20] GoldenFrog. 2018. Privacy Policy. Retrieved 28 February, 2018 from https://www.goldenfrog.com/privacy.
[21] Ah Kioon, M. C., Wang, Z. S. and Deb Das, S. Security analysis of md5 algorithm in password storage. *Applied Mechanics and Materials*, 347 (2013), 2706-2711.
[22] Tancredi, D. 2016. How Apple's Mandatory iOS App Transport Security (ATS) change will affect you. Retrieved 28 February, 2018 from https://appdevelopermagazine.com/how-apple%27s-mandatory-ios-app-transport-security-(ats)-change-will-affect-you/.
[23] Lord, N. The history of data breaches. *Digital Guardian* (2017).