Push Away Your Privacy: Precise User Tracking Based on TLS Client Certificate Authentication

Matthias Wachs, Quirin Scheitle, Georg Carle
Chair of Network Architectures and Services
Technical University of Munich (TUM)
Email: {wachs,scheitle,carle}@net.in.tum.de

***Abstract*—The design and implementation of cryptographic systems offer many subtle pitfalls. One such pitfall is that cryptography may create unique identifiers potentially usable to repeatedly and precisely re-identify and hence track users. This work investigates TLS Client Certificate Authentication (CCA), which currently transmits certificates in plain text. We demonstrate CCA's impact on client traceability using Apple's Apple Push Notification service (APNs) as an example. APNs is used by all Apple products, employs plain-text CCA, and aims to be constantly connected to its backend. Its novel combination of large device count, constant connections, device proximity to users and unique client certificates provides for precise client traceability. We show that passive eavesdropping allows to precisely re-identify and track users and that only ten interception points are required to track more than 80 percent of APNs users due to global routing characteristics. We conduct our work under strong ethical guidelines, responsibly disclose our findings, and can confirm a working patch by Apple for the highlighted issue. We aim for this work to provide the necessary factual and quantified evidence about negative implications of plain-text CCA to boost deployment of encrypted CCA as in TLS 1.3.**

I. Introduction

Encryption offers a wide range of security benefits and is praised by companies to offer security to their customers [11]. A side-effect of cryptography can be the creation of unique cryptographic identifiers, which can, for example, be observed in TLS authentication using X.509 certificates. TLS, including its current version 1.2, sends server and client certificates in plain text before establishing an encrypted channel. Client certificates may contain sensitive user information such as users' real names. Even if these certificates do not leak personal information directly, the client certificates, typically valid and used for several years, establish unique markers for precise identification across repeated observations. In this work, we demonstrate the privacy impact of *Client Certificate Authentication (CCA)* used by Apple's *Apple Push Notification service (APNs)*. With over 1 billion active Apple devices in the world [27], of which many are used by politicians, journalists, or other high-profile groups, this service is relevant in both size and the demographics of its user base. As APNs tries to hold an active connection to its backend servers at all times, tracking users through this connection is promising.

Unique information contained in certificates enable both local and global adversaries with access to network traffic to uniquely identify users, track them over time, and create profiles of their behavior and usage patterns. While existing methods based on traffic correlation or traffic markers only allow to identify users with a certain *probability*, client certificates allow *precise* identification of users or user devices.

In this work, we demonstrate the exploitability of unique CCA identifiers leaked current TLS as used by APNs. We do so by (i) using passive measurements to verify precise re-identification and traceability of certificates in a large metropolitan area network, and (ii) using active measurements to show that APNs logins are globally routed through few central networks, potentially susceptible to a powerful attacker.

Having established both feasibility and impact of this privacy leakage, we follow a responsible disclosure process and report our findings to Apple's Product Security Team. In a very positive exchange, Apple confirmed our findings and its privacy implications. Apple has been diligently working on a patch to resolve this issue, which is included as CVE-2017-2383 in the iOS 10.2.1 and macOS 10.12.3 security updates released on January 23, 2017 [3].

We see the main contributions of our work in:

- Measuring evidence on the privacy and traceability impacts of clear-text CCA
- Proving that a powerful global adversary may easily and passively leverage CCA to track users globally
- Following through the disclosure and patch process to protect the large APNs user base from this vulnerability

We aim for this example to be taken seriously by other applications using TLS CCA, and hope to boost the deployment of encrypted CCA through several strategies, including TLS 1.3, discussed in this paper.

We structure our work as follows: We give background in Section II and discuss related work in Section III. We define the attacker model in Section IV, followed by passive TLS CCA observations in Section V, and a dissection of APNs's global routing in Section VI. We discuss our findings and their disclosure in Section VII, before concluding our work in Section IX.

II. Background

This section gives background on TLS and CCA, push notification services in general, and APNs and its employment of TLS CCA in particular.

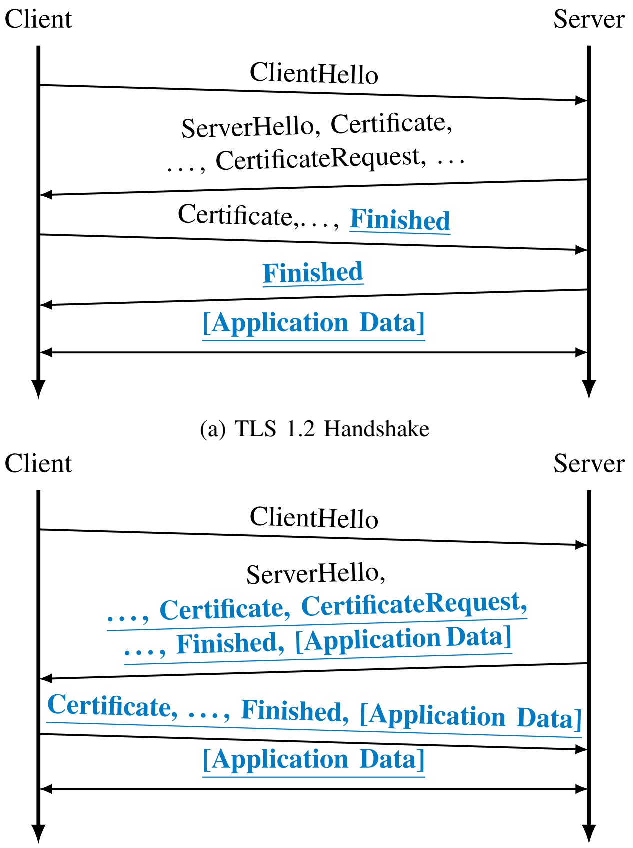
A. TLS & Certificate-Based Authentication

Transport Layer Security (TLS)—often mistakenly still called SSL—is the de facto standard to establish secure communication between systems on today's Internet. The most noticeable benefit of TLS is that it allows to provide secure communications without modifying or impacting higher-layer protocols. The latest version 1.2 of TLS is defined in RFC 5246. The first draft of TLS 1.3 [10] was published in 2014 but standardization is still work in progress. TLS provides mechanisms to authenticate both the destination (server) and the initiator (client) of a connection.

The following explanation focuses on authentication and certificate-related aspects of the TLS 1.2 handshake. When a client establishes a TLS connection, it first sends a *ClientHello* message, containing the cryptographic ciphers supported by the client and other information. The server responds with a *ServerHello* followed by a *Certificate* message containing one or more certificates to authenticate the server. In addition, the server can request a certificate from the client by sending a *CertificateRequest* message. This message may include a list of desired Certificate Authorities (CAs) supported for validation. The client responds with a *Certificate* message containing the client's certificate chain. Only after this unencrypted mutual authentication, both partners establish the desired secure channel. A TLS 1.2 mutual authentication is depicted in Figure 1a.

For authentication, TLS relies on X.509v3 certificates (cf. RFC 5280), asserting that a cryptographic public key belongs to the certificate's *subject*, making certificates identifiable and attributable to users or devices. Certificate validation is performed by the system receiving the certificate. Validation is performed according to the system's requirements, i.e., the systems do not have to use publicly verifiable certificates issued by well-known certificate authorities but can instead rely on private CA infrastructures.

The current draft for TLS 1.3 [10] proposes a different handshake protocol with the specific security goal to protect the endpoints' identities. First of all, a shared secret is established between client and server to protect against passive attackers. The client first sends a *ClientHello* message containing an (EC)DHE key share. The server responds with a *ServerHello* message containing its key share used to compute the shared secret. After such a shared secret is established and communication is encrypted, the server provides the client with its certificate in the *Certificate* message. Only after sending its certificate, it can request a client certificate sending a *CertificateRequest* message. This approach protects both the server and client certificate from observation. The proposed handshake for TLS 1.3 is depicted in Figure 1b. It is important to note that TLS 1.3 can still leak sensitive information since the *ClientHello* (containing the TLS extensions) is still unencrypted. These TLS extensions can, for example, leak the target hostname through the *Server Name Indication (SNI) ClientHello* extension.


Client
Server
ClientHello
ServerHello, Certificate,
. . . , CertificateRequest, . . .
Certificate,. . . , Finished
Finished
[Application Data]
Client
Server
ClientHello
ServerHello,
. . . , Certificate, CertificateRequest,
. . . , Finished, [Application Data]
Certificate, . . . , Finished, [Application Data]
[Application Data]

(a) TLS 1.2 Handshake

(b) TLS 1.3 Handshake

Fig. 1: Handshakes in TLS 1.2 and TLS 1.3 (draft), highlighting unencrypted and **encrypted** data. Both client and server certificates are encrypted in TLS 1.3.

B. Use Cases for Client Certificate Authentication

TLS in combination with CCA is used in a large variety of settings, especially when mutual authentication of communication partners is essential. Where passwords are considered insufficient, CCA can provide a level of multi-factor authentication, as recommended by OWASP [18].

On network level, CCA can be used with network access control and virtual private network authentication. One prominent example is enterprise (wireless) network authentication using 802.1x in combination with the *Extensible Authentication Protocol (EAP)* (cf. RFC 5247). EAP is an extensible authentication framework and supports certificate-based mutual authentication with EAP-TLS (cf. RFC 5216). TLS CCA is also used with OpenVPN [17], one of the most widely used solutions to create virtual private networks (VPN). OpenVPN uses a custom security protocol based on shared keys or TLS. In TLS mode, OpenVPN employs X.509 certificates for mutual authentication.

With HTTPS, TLS can be used to authenticate both the web site and the user. Popular web servers such as Apache, NGINX, and IIS support authentication using CCA. Websites do not widely use CCA as it is complex for users to install and maintain certificates across multiple devices and browsers.

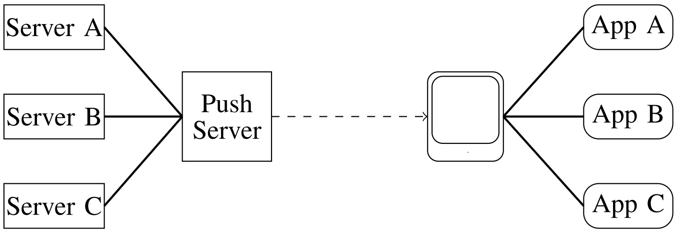


Fig. 2: Push Service Architecture: Messages brokered to Apps through the Push Notification Service.

TLS and CCA are also employed for higher-layer applications such as MQTT (cf. ISO/IEC PRF 20922), a lightweight messaging protocol designed for the "Internet of Things". MQTT can use TLS and CCA for client authentication, but provides no means to authenticate the server.

C. Push Notification Services

Push Notification Services (PNSes) are an essential functionality of modern service ecosystems. PNSes provide a resource efficient approach for service backends to notify (mobile) devices and applications about events. PNSes originate from mobile platforms where resources like energy and network access are limited. All modern mobile platforms are equipped with PNSes, often tightly integrated with the operating system: Apple's APNs on iOS, Firebase Cloud Messaging (FCM) on Google's Android, and Windows Notification Service (WNS) on Windows Phone are prominent examples. PNSes became the most prominent way to notify applications about service events. PNSes are also integrated with desktop operating systems and even web browsers (e.g., Google FCM in Chrome) to efficiently notify applications about service events. This makes PNSes an omnipresent, inevitable link between applications, devices, and service and infrastructure backends. Figure 2 displays this basic concept of PNSes.

When establishing a connection to the PNS, both the device and the third party provider have to authenticate to the service. While for third party provider, authentication approaches like OAuth are used, device authentication is not documented for most services due to their proprietary architecture and tight integration with the operating system. Only Apple documents the use of TLS and CCA with its security guide [5] and APNs documentation [4]. We investigate communication for Google's FCM and Microsoft's WNS, but do not find plaintext client certificates. Google's FCM uses TLS 1.2 and TCP ports 5228 through 5230 to connect to its backend. In our analysis for FCM, no unencrypted client certificates were transmitted with the handshake. The same applies to Microsoft's WNS, which uses TCP port 443 and TLS 1.2.

D. Apple Push Notification service (APNs)

Within Apple's ecosystem, APNs is a key service for the communication between Apple's service backend and user devices and applications. APNs was added to mobile devices with iOS 3.0 in 2009, and extended macOS platforms with the release of Mac OS X 10.7 in 2011. APNs and CCA is also used with iTunes on Windows 7 and later. Support for website notifications in Apple's Safari browser was added with the release of Mac OS X 10.9 in 2013.

On its initial activation, each Apple device is provided with a private cryptographic key and a X.509 certificate, stored in the device keychain. With iTunes, the certificate is created when logging in with an Apple ID and is stored in Window's certificate store. Mutual authentication is performed when the device connects to APNs: Using TLS 1.2 with CCA as described in Section II-A, first the server's certificate is validated by the device. Next, the device sends its client certificate, which is validated by APNs to establish a mutually trusted connection [4].

For *device-to-push-service* communication, APNs uses two different TCP ports [6]: By default the device tries to connect to an APNs server on TCP port 5223. If TCP port 5223 is not reachable (e.g., due to port filtering), APNs connects to TCP port 443 on WLAN only. When connected to both cellular and WLAN or wired networks, APNs prefers cellular data links over WLAN or wired connections [6]. We assume this reduces reconnects and provides a stable link for mobile users.

For its backend, APNs employs a load balancing architecture, where devices connect to one of 50 APNs DNS records named *[1-50]-courier.push.apple.com* [6]. These names are served by Akamai's DNS service and mapped to *[1-50]-.courier-push-apple.com.akadns.net* using CNAME records. These resolve to a geographically close name, for example, *pop-eur-central-courier.push-apple.com.akadns.net* when resolved from Munich. These names resolve to a variety of IP addresses in Apple's 17.0.0.0/24 address range.

III. Related Work

To the best of our knowledge, only limited work on the privacy implications of digital certificates with a particular focus on certificate-based client authentication exists. In particular, no assessment of real world implications of CCA exists.

Parsovs [19] gives an extensive overview of operational problems related to TLS CCA. He rather addresses implementation details than general privacy implications, but also highlights that client certificates should not be transmitted in clear text. He mentions TLS modifications proposed for standardization that ensure encrypted transport of client certificates, but notes that a modification of the TLS standard would take years to be adopted. He explores how renegotiation, as a possible workaround for an encrypted transmission of client certificates, has a negative impact on performance. Aura and Ellison [7] analyze privacy implications of certificates and certificate systems in detail, and highlight identity leakage and unique keys as main issues with certificates. The authors describe several anonymity techniques such as key-oriented access control and certificate reduction as solutions to these problems, and how the SDSI PKI, a distributed PKI proposed by Rivest and Lampson in [23], leverages these techniques. For X.509, these approaches were never adopted.

Chung et al. [9] recently tracked over 5M devices for more than one year by actively scanning for invalid X.509

certificates. In contrast to our study, active scans can not discover TLS client certificates. Also, they rather find servers, such as home routers or storage devices, than mobile client devices.

In an Internet-Draft [21], Ray highlights the issues of TLS's unencrypted handshake and its implications with the goal to extend TLS with an encrypted handshake. Ray proposes to establish a secure connection first by exchanging (EC)DH parameters and only then to transmit certificates using a second *ClientHello* and *ServerHello* command. The draft expired in May 2012 without being adopted.

Langley authored an Internet-Draft [13] aiming to extend the TLS handshake with an encrypted certificate. He proposes to first establish an encrypted connection using the *ChangeCipherSpec* command and only exchange certificates after this step. This approach introduces incompatibilities into the TLS protocol. The draft was not adopted and expired in April 2012.

In literature, unique identifiers and certificate based authentication in particular are long known to create issues for user privacy. Nonetheless, neither were these issues addressed in the TLS standard up to TLS 1.2, nor was the current TLS 1.2 standard modified, nor was one of the suggested approaches adopted to counteract this issue. While TLS 1.3 addresses this problem, full adoption of TLS 1.3 by services and clients will take a long time with user privacy being at risk. To the best of our knowledge, no scientific work tried to analyze or quantify the impacts of CCA on user privacy and the possibility for user tracking, especially in combination with always-connected mobile personal devices.

Besides cryptographic traceability, unique device identifiers have attracted privacy concerns and mitigation actions: Mobile devices used to be well traceable through WLAN MAC addresses. This was mitigated years ago by Apple and other vendors by offering random MAC addresses to unknown networks [28], though not always effectively, as recent work indicates [14]. With IPv6, stateless autoconfiguration calculates an IPv6 address based on the network interface's MAC address, with the threat of global traceability of devices. To protect against such tracking, IPv6 privacy extensions (cf. RFC 4941) randomly select IPv6 addresses.

There exists a large body of work on tracking users through DNS [12], Mobile Apps [22], Browser Fingerprinting [1] and many other means. We consider these approaches different from ours in that they are either (i) stochastic, i.e., not leveraging cryptographically unique fingerprints, or (ii) active, i.e., requiring attacker capabilities beyond listening.

IV. Attacker and Threat Model

We define three types of attackers with different motivations and capabilities. These attackers are specific points in a possible spectrum of attackers.

Attacker (a) is a powerful entity interested in precisely tracking and identifying users globally. It is modeled after a nation state or a member of the intelligence community. This attacker can read network traffic of one or several large Internet backbone networks and/or Internet Exchange Points (IXPs). In addition, this attacker can typically coerce network operators to provide extended information about users, for example, by mapping IP addresses to network locations.

Attacker (b) is a local or regional entity with access to a single large network. It could typically be an individual organization or company, with an interest to track roaming and usage patterns of users over different parts of the network.

Attacker (c), the operator of a small network, could also enhance its tracking capabilities by the use of TLS client certificates, but can already precisely track typical counts of <10 users through device names or MAC addresses. We hence exclude attacker (c) from further analysis.

Currently, attackers (a) or (b) typically will resort to stochastic or opportunistic "markers" to identify people. For example, in the NSA's QuantumInsert program, some markers ("realms") used are Facebook, HotMail, or YouTube cookies as well as public static IPv4 addresses [16, p. 5]. In contrast to the markers mentioned above, we highlight that client certificates used by TLS CCA are cryptographically unique, used by different services, rarely change over time, are wide-spread on billions of devices, and are frequently transmitted, especially when used with mobile devices and always connected services, such as mobile push notification services.

We highlight that attackers (b) and (c) typically also have other possibilities of tracking users within their own networks, based on, for example, MAC addresses or, for cellular operators, IMEI identifiers. We also emphasize that tracking users based on TLS Client Certificates requires only passive network access, enabling attacks to be conducted (i) in a lightweight manner, and (ii) retrospectively on stored network samples.

V. Verification of Metropolitan Area Network Traceability

This section emulates attacker (b) as defined in Section IV, which has interception capabilities in a metropolitan area network. We verify that this attacker can effectively track users through passive measurements.

A. Experimental Setup and Methodology

To give a detailed analysis and evaluation of APNs communication patterns and to be able to analyze the impact of APNs's use of CCA on user privacy in the metropolitan area network attacker scenario, we conduct an extensive analysis of APNs communication. This scenario does not include mobile cellular connections, but only WLAN and wired connections.

We monitor APNs communication at the Internet uplink of the Munich Scientific Network (MWN), a metropolitan area network connecting about 100,000 users to the Internet. We consider the MWN an insightful environment as it provides properties of scientific (connecting several large universities), corporate (employees at various organizations), and residential (student residence halls) networks.

Using *tcpdump*, we capture APNs TLS handshake packets containing certificates over the course of 17 days. We employ a *pcap* filter for TCP traffic with destination ports 443, 5223, 2195, and 2196 used by APNs as described in Section II-D.

The filter also limits capturing to TLS handshake protocol messages (*Record Layer ContentType = handshake*) containing certificate information (*Handshake Type = Certificate*).

We capture data as *pcap* files and use a python processing tool leveraging *scapy* [24] extended with *scapy-ssl_tls* [25] to extract certificate information. Using this tool, we extract the following information from the *pcap* files:

- **Connections:** timestamp, source IP address, source port, destination IP, destination port, certificates found
- **Certificates:** X.509 version, serial number, subject, issuer, public key modulus, public key size, start and end of validity, fingerprint, extensions contained
- **Certificate extensions:** name/OID and values

While processing, we perform DNS reverse resolution for source and destination IP addresses to obtain DNS hostnames.

B. Ethical Considerations

We follow an internal multi-party approval process before any measurement activities are carried out. This approval process incorporates the proposal of Partridge and Allman [20] to assess whether the collection of data can harm individuals and whether the collected data reveals private information.

The focus of this work is solely to create profiles of devices. We do not attempt create a link between devices (i.e., the certificates found) and the respective owner's identity, except five members of our research group giving informed consent to case studies. Information contained in the certificates does not provide a possibility to directly link it with a user. We neither try to link certificates to users based on identifiers such as IP addresses, nor try to uncover their geographical location.

The measurements conducted for this work were performed on an isolated measurement infrastructure not accessible from the Internet. Data obtained in our measurements must remain on this infrastructure. The methodology for this experiment was thoroughly documented before the measurement was conducted. The experiment's methodology fully complies with the strict code of conduct required by our internal ethical review processes, and we obtained approval for our measurements in this process. We hence conclude it is ethical to conduct the experiment, but will, in contrast to our usual policy, not share raw data from this work with the public.

C. Properties of APNs Certificates

Our experiments show that APNs, relying on standard TLS 1.2, employs standard X.509v3 certificates for mutual authentication between devices and APNs servers. APNs devices use 1024 bit RSA keys and are provided with certificates issued by an *Apple iPhone Device CA* certificate authority using the common name `C=US, O=Apple Inc., OU=Apple iPhone, CN=Apple iPhone Device CA`.

Our investigations show that certificates of mobile devices (iPhones, iPads) running iOS have different properties than certificates of desktop devices running macOS and iTunes on Windows. This allows us to distinguish between mobile and desktop devices: certificates for mobile devices have a subject containing a unique identifier and CA information similar to `CN=7F3F3123-1234-4EF8-5678-F3CDE236E1EF, C=US, ST=CA, L=Cupertino, O=Apple Inc., OU=iPhone`. In contrast, certificates for desktop devices only contain the unique identifier: `CN=7F3F3123-1234-4EF8-5678-F3CDE236E1EF`. By investigating several devices of consenting users, we verify that the certificate's *"not valid before"* timestamp reflects the precise time of device registration with Apple. The certificate validity for iOS devices is 3 years, while certificates for desktop devices only have a validity of 1 year.

TABLE I: Number of observed TLS client certificates for five most frequent Issuer Distinguished Names.

#Certs	Issuer Distinguished Name
56128	/C=US/O=Apple Inc./OU=Apple iPhone/CN=Apple iPhone Device CA
334	/CN=Layer Client CA/C=US/L=San Francisco/O=Layer, Inc/ST=CA
221	/CN=AnyDesk Client
76	/C=KR/ST=Kyunggido/L=Suwon/O=Samsung Electronics (*redacted*)
52	/CN=Ricoh Remote Service (*redacted*)

D. Data Capturing and Basic Statistics

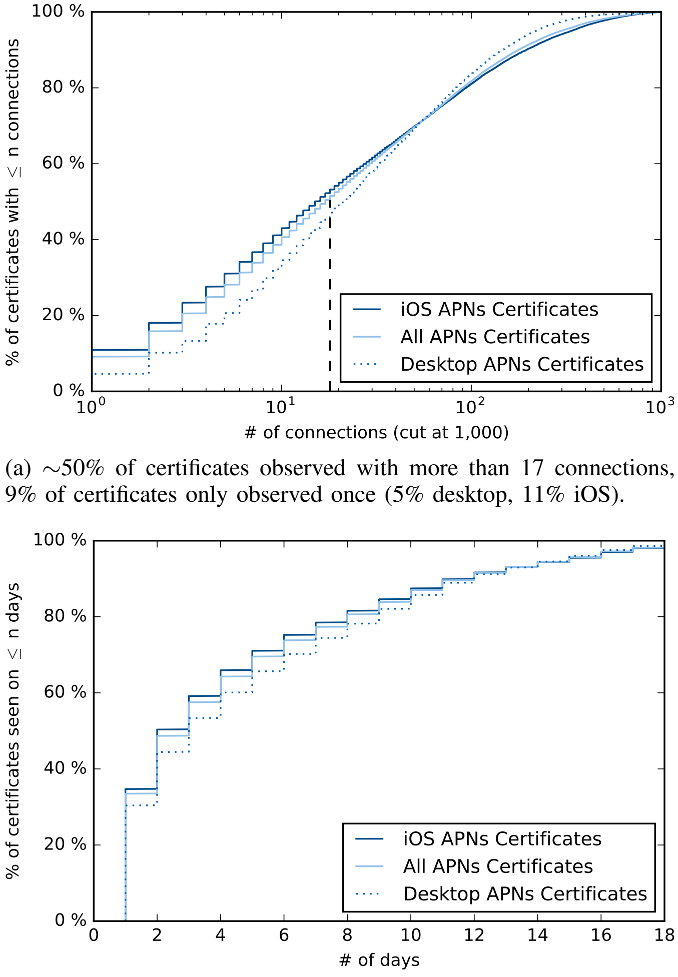
Capturing for 17 days in September 2016, we observe 70,173,492 TLS CCA connections with 57,477 unique client certificates, of which 56,128 (97%) stem from APNs. Across all 57,477 client certificates, we find 220 distinct issuer distinguished names (*DNs*). The Top 5 issuer *DNs* are shown in Table I, exhibiting a clear dominance of APNs certificates in our data set, but also showing other use cases of CCA such as client service authentication and remote device management.

The 56,128 APNs certificates can further be broken down into 40,313 (72%) iOS certificates and 15,815 (28%) desktop certificates. Of the 70,173,492 TLS CCA connections, we identify 68,231,915 as APNs certificates on port 5223 and 1,588,864 as APNs certificates on port 443. The small remainder of 352,713 connections did not use APNs certificates.

E. Recurrence of Certificates

To assess whether the described capture would allow for reliable traceability of users, we first investigate the recurrence of certificates: Only certificates observed more than once allow for a basic level of user tracking. Figure 3a displays the number of connections observed per certificate.

We display total number as well as a breakdown into iOS and desktop devices. We find about 50% of total certificates to have 17 or more connections. The fact that only $\leq$10% of certificates are only observed once speaks to traceability of individual certificates. Please note that 95% of desktop certificates were observed more than once, as desktop devices typically can not connect to APNs through cellular service, hence always use our observed WLAN or wired connections to connect. We find some certificates with dozens of millions of connections, which we can link to an iOS continuous integration build cluster within the network. Figure 3b displays the number of days that individual certificates were observed to connect on. About 50% of certificates connected on 3 or more days. We consider these regular users of our network, which are likely well traceable over longer periods.



(a) ~50% of certificates observed with more than 17 connections, 9% of certificates only observed once (5% desktop, 11% iOS).

(b) 50% of certificates observed on 3 or more separate days, 34% only observed on 1 day (30% desktop, 35% iOS).

Fig. 3: Statistics on certificate recurrence — *Click on any data-driven figure in this paper to go to its source code*

F. Deductions about User Behavior

We next set out to explore the level of insight that can be deduced about individual users and devices.

Figure 4 tracks the APNs certificate of one of the authors' MacBook to ensure applicability of our methodology and highlight the relevance of our work. Based on the subnet of the source IP address, we can track whether this user logged in from his desk, through WLAN, or through VPN. Days off and days with outside meetings can clearly be identified, for example, the first Monday, with a remote VPN login followed by desk presence later that day. Meetings away from desk are visible through WLAN logins, for example, the second Wednesday and Thursday. This exhibits the power of tracking users from APNs certificates at a metropolitan area network level. This information is also accessible to eavesdroppers on the path between this metropolitan area network and the APNs backend. The mapping of IP addresses to certain networks and characteristics may seem difficult for outsiders, but is eased by descriptive reverse DNS names deployed in these networks.

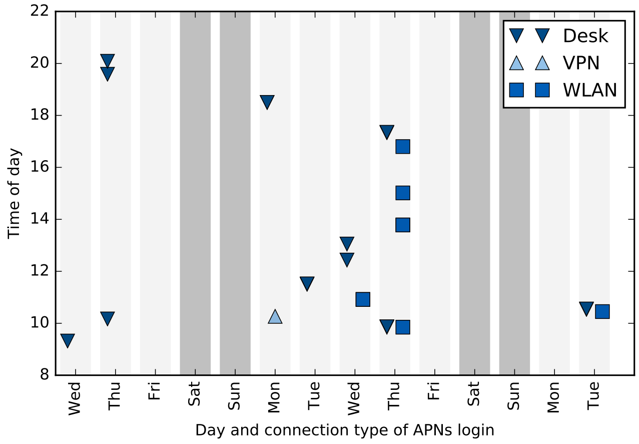


Fig. 4: User Study: APNs logins clearly show work starting times and locations (derived from subnet data).

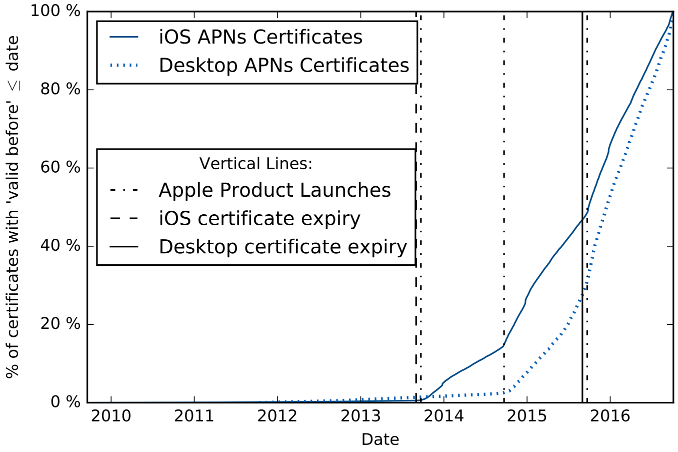


Fig. 5: CDF of "*not valid before*" timestamps of APNs certificates. Highlighted are the influence of Apple's end of September product launches as well as the expiry threshold dates for iOS certificates, which are valid for three years, and desktop certificates, which are valid for one year.

G. Deductions about Devices

As the certificate's *"not valid before"* timestamp reflects the precise time of device registration with Apple, this can narrow down the device type, as Apple typically stops selling previous hardware models with the availability of new devices. This means that the majority of certificates created at a certain point of time will belong to devices from the Apple hardware offerings at that time.

Figure 5 depicts the cumulative distribution of the "*not valid before*" timestamps in observed APNs certificates. Several observations can be taken from this figure:

First, certificates are used beyond their expiry date. As certificates are valid for 3 years for iOS and 1 year for desktop devices, we can plot the expiration threshold of certificates in Figure 5. Certificates issued before these thresholds have exceeded their validity period.

We can see that 28% of desktop and 1% of mobile devices use expired certificates. We conclude that APNs clients do not

TABLE II: Eavesdropping on just 10 networks allows to follow APNs messages of over 80% of users globally or nationally.

Rank	Global		Germany	
	IXP/AS	Σ Paths	IXP/AS	Σ Paths
1	AS3356 (L3)	25%	IXP DE-CIX	30%
2	AS1299 (Telia)	40%	AS3320 (DTAG)	52%
3	AS174 (Cogent)	54%	IXP E-CIX	60%
4	AS7922 (Comcast)	61%	AS6830 (Liberty)	68%
5	AS6830 (Liberty)	65%	AS31334 (VF/Kabel D)	74%
6	AS4637 (Telstra)	69%	AS1273 (C&W)	77%
7	AS6453 (Tata)	72%	AS3356 (L3)	80%
8	AS2828 (XO)	75%	AS680 (DFN)	83%
9	AS12322 (Free)	78%	AS34419 (VF Group)	85%
10	AS3320 (DTAG)	81%	AS6805 (Telefonica)	88%

systematically renew their certificates and only re-installations or re-registrations cause certificate renewals over time. This lack of systematical certificate renewal extends the possible duration a device can be traced. Furthermore, Apple's product presentations in end of September typically boost registration of new devices and therefore certificates.

VI. Global Routing of APNs

In this chapter we investigate whether a powerful attacker, as described as attacker (a) in Section IV, can effectively track individual users by eavesdropping on central Autonomous Systems (ASes) or Internet Exchange Points (IXPs).

Based on Apple's documentation [6] and confirmed by measurement, we establish that (i) APNs uses IP addresses from the *17.0.0.0/8* prefix and (ii) devices resolve one of *[1-50]-courier.push.apple.com* to connect to an individual APNs server. We globally resolve those DNS names through RIPE Atlas and find them to redirect, based on resolver location, into several regional Akamai clusters (located in Apple's *17.0.0.0/8* IP range), featuring a total of 69 subnets of size /24. In the next step, we randomly pick one of the observed IP addresses in each of the 69 /24 subnets, resulting in 69 measurement targets. We then conduct in-protocol (using TCP/5223) *traceroute* measurements towards each of the 69 targets, selecting 1000 random globally distributed probes for each measurement. Over all 69 global traceroute measurements, we use a total of 1959 RIPE Atlas probes, located in 1115 ASes and 115 countries (according to probe properties). Using traiXroute [15] and CAIDA's AS mappings [8], we map the IP addresses observed on the *traceroute* paths to IXPs and ASes. Next, for every IXP or AS, we count the number of *traceroute* measurements that it is present in. We also conduct these steps on a German subset of APNs servers and RIPE Atlas probes to compare global and nation-centric views.

The top 10 ASes and IXPs for both global and nation-centric views are shown in Table II. Table II confirms that eavesdropping capabilities in just 10 ASes or IXPs will allow an attacker to eavesdrop over 80% of our traces.

One might question whether our methodology of using 1959 RIPE Atlas probes is a fair sample of the global APNs user population. We argue that as RIPE Atlas probes are generally well distributed across countries and networks (unlike, for example, PlanetLab's focus on academic networks), a large random sample of RIPE Atlas probes represents a fair approximation of global Internet traffic sources. Furthermore, we are not aware of any reason to assume that the APNs population is significantly different from typical Internet traffic sources.

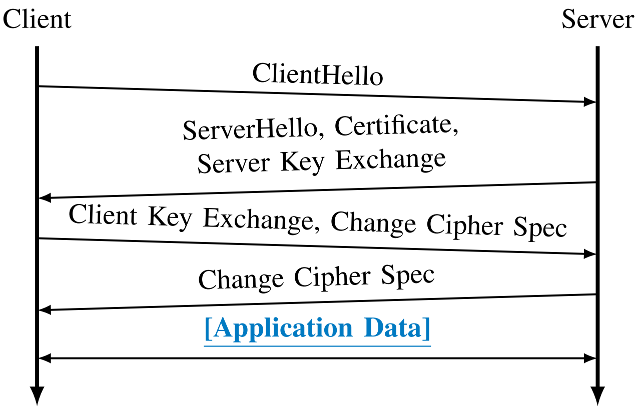


Fig. 6: Improved APNs v3 using TLS1.2, postponing CCA into the **encrypted** Application Layer Data Exchange

Based on this we consider the hypothesis of our inquiry, that an attacker with access to few core networks can track users across many access networks, holds true even against possible distortions from a sample bias.

VII. Responsible Disclosure & Mitigation

We disclosed the privacy issues of CCA-based APNs authentication to Apple's Product Security Team in October 2016. Apple's Product Security Team reacted within a day and quickly acknowledged this issue and its severity. Apple immediately started patch development and held contact with us through several calls to solve this issue and allow us to test the proposed fix.

Despite the complexity of developing and testing this fix in iOS, macOS, iTunes for Windows, and the APNs server infrastructure, the patch was already included in the public iOS and macOS security updates released in January 2017. We tested the patch using the public beta on macOS 10.12.3 Beta 4 (16D30a) and iOS 10.2.1 Beta 4 (14D27). We find the issue discussed in this paper resolved through a changed usage of TLS 1.2, moving the client authentication into the encrypted Application Layer Data Exchange, as depicted in Figure 6. The improved version is indicated in the *ClientHello* message with an Application Layer Protocol Negotiation (ALPN) Next Protocol *apns-security-v3* (opposed to *apns-security-v2* for the version with the vulnerability described in this paper).

VIII. Discussion

While negative privacy implications of clear-text certificate transmission in the TLS handshake have been publicly discussed before, the impact of clear-text CCA has never before been empirically quantified against a specific and realistic threat model. This might have been one of the reasons why remediation of this problem was not a priority in the many years of TLS usage and has been delayed to the major rehaul of the TLS handshake with TLS 1.3.

With this work, we provide evidence that clear-text TLS CCA may have crucial impact on user privacy. Based on empirical measurements, we show that this impact is easily abusable by the attackers defined in our threat model. We follow a responsible disclosure approach and find Apple to assess our insights and their implications severe enough to start immediate and expedited patch development for billions of affected devices.

Generalization of Results: With our passive measurements taking place at one specific WLAN and wired network, the question arises whether its results generalize to a more heterogeneous network that also includes cellular connections and accompanying Carrier-Grade NATs (CGNs) or other middleboxes. We strongly argue that our results generalize to those networks. For geographical tracking of users through passive observations of APNs handshakes, the network operator must be able to infer a user location based on externally visible properties of the handshake, typically the source IP address. We argue that operators of all kinds of networks will have this capability, as it is typically required by law in most countries to resolve abuse and other inquiries. This capability also enables a global adversary to locate users precisely: By coercing local network operators to pin-point users based on their public IP address, powerful global adversaries can leverage TLS CCA to globally and precisely track users. Even without such a collaboration CCA-based tracking is useful for an attacker, as often not the precise geographical location is required by the attacker, but a more coarse localization is sufficient, for example, if a user visited a particular country or region. Such an approximate localization can still be obtained with approaches such as CGNs in place. Aside from limited geographical accuracy, an attacker can still learn about user behavior and infer usage patterns from temporal correlation.

Remediation Strategies: We consider the elimination of clear-text TLS CCA from current applications an important vector to enhance privacy in networked systems and discuss several strategies for doing so: First and foremost, the use of TLS 1.3 is likely the preferred way for most applications. However, TLS 1.3 standardization is not finished to date, and roll-out to a critical mass of devices may take several years. To mitigate the impact, applications might look to reduce the frequency of TLS CCA submissions by aggressively leveraging TLS session resumption. However, TLS session tickets might by themselves create identifiable patterns. In a short-term strategy, applications may delay the client certificate submission to the encrypted application layer, but this requires major implementation effort on client and server side. Therefore, this approach may only be feasible in centralized architectures such as APNs. Also, changes to a username/password approach may be feasible, but usually come with usability downsides. Another strategy to reduce the number of observable TLS CCA transmissions, as already employed by APNs, is the prioritized use of long-lived and stable networks (e..g, cellular links) over typically transient wired and WLAN connections.

IX. Conclusion and Future Work

With this work we present—to the best of our knowledge—the first qualitative and quantitative assessment of CCA privacy implications. We document the use of TLS Client Certificate Authentication (CCA) by Apple's Push Notification Service (APNs). As TLS transmits clients certificates in clear text, the frequent logins of devices at the APNs backend provide opportunity for precise user tracking through highly unique cryptographic properties of client certificates. We validate our claim by a 17-day passive capturing at the uplink of a major scientific network where we can spot and track more than 56,000 APNs certificates. We demonstrate how certificate information is suited to track individual users and derive device information. To display that this tracking technique would be feasible for a powerful eavesdropper, we show through global measurements that access to only 10 networks may provide opportunity to track APNs users in over 80% of access networks, both globally and nationally. We highlight that we do not foremost see this as a vulnerability of APNs, but rather a weakness in the TLS 1.2 protocol, and strongly support the encrypted transmission of client certificates in TLS 1.3. We hope that the quantification of impact in this study helps to accelerate the adoption of TLS 1.3.

Data Release: In [26], we outline our aim for repeatable, replicable and reproducible research as defined by ACM [2]. With ethics-driven exceptions discussed below, we publish all data and source code used to create this publication under

`https://github.com/tumi8/cca-privacy`

This includes code to create data-driven figures, which is also referenced as a clickable hyperlink for each data-driven figure in this work. As discussed in Section V-B, we can not provide the passively captured APNs certificates for ethical reasons. However, we publish selected captures that highlight the issues and its solution. For figures that build on private data, we provide anonymized datasets. In addition to our published data, our RIPE Atlas measurements will also be long-term accessible through RIPE Atlas.

Future Work: We plan to further quantify the impact of our research by periodically measuring the amount of clear-text TLS CCA observable, and, specifically, the distribution of fixed APNs versions. Also, the adoption of TLS 1.3 raises interesting and quantifiable questions. Furthermore, the leakage of sensitive information through unencrypted TLS extensions may be an interesting research field. Finally, we consider the identification of cryptographically unique identifiers in other authentication protocols an important research goal.

Acknowledgments: We thank the Leibniz Supercomputing Centre (LRZ) of the Bavarian Academy of Sciences (BAdW) for their support in conducting our verification. We thank Apple's Security Team for the good and trustful collaboration, their valuable feedback, and their efforts to improve security and privacy on the Internet. This work has been supported by the German Federal Ministry of Education and Research, project X-CHECK, grant 16KIS0530.

REFERENCES

[1] G. Acar, C. Eubank, S. Englehardt, M. Juarez, A. Narayanan, and C. Diaz. The web never forgets: Persistent tracking mechanisms in the wild. In *Conference on Computer and Communications Security (CCS)*. ACM, 2014.

[2] ACM. Result and Artifact Review and Badging. https://www.acm.org/publications/policies/artifact-review-badging, Acc. Jan 18 2017.

[3] Apple. About the security content of ios 10.2.1. https://support.apple.com/en-au/HT207482, Acc. March 28, 2017.

[4] Apple Developer Documentation. Apple Push Notification Service. https://developer.apple.com/go/?id=push-notifications, Acc. Oct 7 2016.

[5] Apple Inc. iOS Security. https://www.apple.com/business/docs/iOS_Security_Guide.pdf, Acc. Jan 12 2017.

[6] Apple Inc. TN2265 - Troubleshooting Push Notifications. https://developer.apple.com/library/content/technotes/tn2265/_index.html, Acc. Oct 7 2016.

[7] T. Aura and C. Ellison. Privacy and Accountability In Certificate Systems. Technical report, Helsinki University of Technology, 2000.

[8] Center for Applied Internet Data Analysis. Routeviews Prefix to AS mappings Dataset. http://www.caida.org/data/routing/routeviews-prefix2as.xml, Acc. Sep. 28, 2016.

[9] T. Chung, Y. Liu, D. Choffnes, D. Levin, B. M. Maggs, A. Mislove, and C. Wilson. Measuring and Applying Invalid SSL Certificates: The Silent Majority. In *ACM Internet Measurement Conference*. ACM, 2016.

[10] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.3. Internet draft, work in progress, https://tools.ietf.org/html/draft-ietf-tls-rfc5246-bis-00, 2014.

[11] N. Gibbs. Tim Cook Interview in Time Magazine. http://time.com/4261796/tim-cook-transcript/, Acc. Oct 7 2016.

[12] S. Krishnan and F. Monrose. DNS Prefetching and Its Privacy Implications: When Good Things Go Bad. In *USENIX Conference on Large-scale Exploits and Emergent Threats: Botnets, Spyware, Worms, and more*. USENIX Association, 2010.

[13] A. Langley. Transport Layer Security (TLS) Encrypted Client Certificates. https://tools.ietf.org/html/draft-agl-tls-encryptedclientcerts-00, 2016.

[14] J. Martin, T. Mayberry, C. Donahue, L. Foppe, L. Brown, C. Riggins, E. C. Rye, and D. Brown. A Study of MAC Address Randomization in Mobile Devices and When it Fails. *arXiv:1703.02874*, 2017.

[15] G. Nomikos and X. Dimitropoulos. traIXroute: Detecting IXPs in Traceroute Paths. In *Passive and Active Measurement - 17th International Conference, PAM*. Springer, 2016.

[16] NSA. Quantum Insert. https://www.eff.org/files/2014/01/02/20131230-spiegel-tao_quantum_tasking.pdf, Acc. Jan 12 2017.

[17] OpenVPN. OpenVPN cryptographic layer. https://openvpn.net/index.php/open-source/documentation/security-overview.html, Acc. Jan 18 2017.

[18] OWASP Foundation. Authentication Cheat Sheet. https://www.owasp.org/index.php/Authentication_Cheat_Sheet#Consider_Strong_Transaction_Authentication, Acc. Dec 20 2016.

[19] A. Parsovs. Practical Issues with TLS Client Certificate Authentication. In *Network and Distributed System Security Symposium (NDSS)*. The Internet Society, 2014.

[20] C. Partridge and M. Allman. Ethical considerations in network measurement papers. *Communications of the ACM*, 2016.

[21] M. Ray. Transport Layer Security (TLS) Encrypted Handshake Extension. https://tools.ietf.org/html/draft-ray-tls-encrypted-handshake-00, 2016.

[22] J. Ren, A. Rao, M. Lindorfer, A. Legout, and D. Choffnes. ReCon: Revealing and Controlling PII Leaks in Mobile Network Traffic. In *International Conference on Mobile Systems, Applications, and Services*, MobiSys '16, New York, NY, USA, 2016. ACM.

[23] R. L. Rivest and B. Lampson. A Simple Distributed Security Infrastructure (SDSI). http://groups.csail.mit.edu/cis/sdsi.html, Acc. Jan 25 2017.

[24] Scapy. http://www.secdev.org/projects/scapy, Acc. Aug 13 2017.

[25] Scapy SSL TLS. https://github.com/tintinweb/scapy-ssl_tls, Acc. Aug 15 2017.

[26] Q. Scheitle, M. Wählisch, O. Gasser, T. C. Schmidt, and G. Carle. Towards an Ecosystem for Reproducible Research in Computer Networking. In *ACM SIGCOMM 2017 Reproducibility Workshop*.

[27] N. Statt. 1 Billion Apple Devices are in Active Use around the World. http://www.theverge.com/2016/1/26/10835748/apple-devices-active-1-billion-iphone-ipad-ios, Acc. Oct 7, 2016.

[28] M. Vanhoef, C. Matte, et al. Why MAC Address Randomization is not Enough: An Analysis of Wi-Fi Network Discovery Mechanisms. In *Asia Conference on Computer and Communications Security (ASIACCS)*. ACM, 2016.