CPV: Delay-based Location Verification for the Internet

AbdelRahman Abdou, *Student Member, IEEE,* Ashraf Matrawy, *Senior Member, IEEE,* and P.C. van Oorschot, *Member, IEEE*

Abstract—The number of location-aware services over the Internet continues growing. Some of these require the client's geographic location for security-sensitive applications. Examples include location-aware authentication, location-aware access policies, fraud prevention, complying with media licensing, and regulating online gambling/voting. An adversary can evade existing geolocation techniques, e.g., by faking GPS coordinates or employing a non-local IP address through proxy and virtual private networks. We devise *Client Presence Verification* (CPV), a delay-based verification technique designed to verify an assertion about a device's presence inside a prescribed geographic region. CPV does not identify devices by their IP addresses. Rather, the device's location is corroborated in a novel way by leveraging geometric properties of triangles, which prevents an adversary from manipulating measured delays. To achieve high accuracy, CPV mitigates Internet path asymmetry using a novel method to deduce one-way application-layer delays to/from the client's participating device, and mines these delays for evidence supporting/refuting the asserted location. We evaluate CPV through detailed experiments on PlanetLab, exploring various factors that affect its efficacy, including the granularity of the verified location, and the verification time. Results highlight the potential of CPV for practical adoption.

Index Terms—Location-aware Authentication; Location-based Services; Location-verification; Internet Measurements; Geolocation

1 Introduction

Over the Internet, *Location-Sensitive Providers* (LSPs) are those that customize their content/services based on the geographic locations of their *clients* (the software that communicates with the LSP, typically a web-browser). Some LSPs restrict their services to certain geographic regions, such as media streaming [2] (e.g., hulu.com); others limit certain operations to a specific location, such as online voting (e.g., placespeak.com), online gambling (e.g., ballytech.com), location-based social networking [3] (e.g., foursquare.com), or fraud prevention (e.g., optimalpayments.com). LSPs may also use location information as an additional authentication factor to thwart impersonation and password-guessing attacks (e.g., facebook.com). Privacy laws differ by jurisdiction, which allows/bans content based on region [4]. The nature of the provided services may motivate clients to forge their location to gain unauthorized access.

Existing geolocation technologies, commonly used in practice, are susceptible to evasion [5]. For example, the W3C geolocation API [6] defines an interface that allows the client's web browser to determine and return the client's location to the requesting LSP. Browser vendors usually rely on common location-determination technologies, such as Global Positioning System (GPS) [7] or WiFi Positing System (WPS) [8]. Because the client sends its location to the LSP, it can submit forged location information [3]. Tabulation-based techniques, where a geolocation service provider maintains tables that map IP addresses to locations—e.g., MaxMind [9], can be evaded through IP address-masking technologies [10] such as proxy servers and anonymizers [11]. Geolocation that is based on active delay measurements [12], [13] is prone to an adversary corrupting the delay-measuring process [14]. A location verification technique is therefore required to provide greater assurance of the veracity of the specified location.

Various solutions have been proposed to verify location claims in wireless networks [15], [16]. However, solutions in this domain cannot be directly adopted by multi-hop networks, e.g., the Internet, due to delay characteristics of different domains. For example, Internet delays are stochastic [17], whereas in single-hop wireless networks, delays can be estimated from the distance the signal spans and the speed of its propagation.

Verifying the location of Internet clients is a challenging problem [5]. A practical approach must address critical challenges such as handling of IP address-masking, and ensuring the correctness of location information submitted by the client. We present and evaluate *Client Presence Verification* (CPV), a delay-based technique designed to verify a client's geographic location. Experimental results show that CPV can achieve a granularity equivalent in area to a circle with radius $\sim 400km$.[1] CPV is designed to resist known geolocation-circumvention tactics as it (1) does not rely on the client's IP address, (2) does not rely on client-submitted information, and (3) is designed such that manipulating the delays is not in the dishonest client's favor. CPV takes as an input the client's asserted location, and outputs a number between 0 and 1 (inclusive) representing its confidence of the asserted location. Under appropriate calibration, the output can then be translated to an accept/reject decision.

- *A. Abdou is with the Department of Systems and Computer Engineering, Carleton University. Email: abdou@sce.carleton.ca*
- *A. Matrawy is with the School of Information Technology, Carleton University. Email: ashraf.matrawy@carleton.ca*
- *P.C. van Oorschot is with the School of Computer Science, Carleton University. Email: paulv@scs.carleton.ca*

Version created on: June 30, 2015. This work extends a preliminary version presented at the IEEE conference on Communications and Network Security (CNS 2014) [1]. The final version of this paper will appear in the IEEE Transactions on Dependable and Secure Computing (TDSC). This is the authors' copy for personal use.

1. The verification region is in fact a triangular, rather than a circular, one as we explain in §4.

A common challenge faced by delay-based geolocation techniques is to find an accurate delay-to-distance mapping function, and thus factors affecting the correctness of this mapping have been well studied in the literature [18], [19]. CPV undertakes a set of measures to mitigate the effect of these factors. For example, it mitigates path asymmetry [20] by using a novel protocol,[2] to deduce one-way delays (OWDs) to/from a potentially dishonest client. Additionally, CPV mitigates network instability [22] by iterating the delay-measurement process.

We analyze the effect of several factors on the correctness of CPV by evaluating its false reject and false accept rates using PlanetLab [23]. For example, results show that the farther an adversary's true location is from the asserted (fraudulent) location, the more likely it is for CPV to correctly reject this assertion; CPV correctly rejected 97% (1,749 of 1,803) of fraudulent location assertions that were $>200km$ away from the adversaries' true locations. We then discuss how CPV mitigates tactics that evade common geolocation techniques, and other potential CPV-specific tactics.

We make the following contributions:

1) Devising a novel protocol to estimate, at a given time, the forward and reverse OWDs between two hosts over the Internet (§5). The protocol can give more accurate OWD estimates than half the round-trip time (RTT), while requiring less cooperation between the two hosts than commonly required by OWD-estimation protocols [24].
2) Presenting CPV, an approach for verifying location assertions of clients over the Internet. CPV leverages the OWD-estimation protocol noted above, and uses heuristics that improve the accuracy of delay-to-distance mapping. To the best of our knowledge, CPV is the first algorithm that uses delays to verify (rather than determine) client locations over the Internet.
3) Evaluating CPV through detailed experiments on PlanetLab, with nodes based in the United States and Canada. The evaluation involves analyzing the algorithm's efficacy in distinguishing honest clients from others, and other factors that affect the accuracy of CPV's results.

The rest of this paper is organized as follows. §2 provides a summary of the literature on delay behavior over the Internet, and its relationship to geographic distances. The threat model is discussed in §3, and CPV is explained in §4. §5 presents the novel OWD-estimation protocol that CPV uses. An empirical evaluation of CPV and a security discussion are presented in §6 and §7 respectively. Related work on location verification is discussed in §8. §9 concludes.

2 Background

Delay characterization between Internet hosts plays a prominent role in numerous applications such as distributed web-caching, server placement in Content Distribution Networks, clock synchronization, overlay Peer-to-Peer networks, Internet geolocation, application-layer mutlicast, and timeout estimations in TCP. Due to inherent sensitivity, factors affecting delays between Internet nodes have been well studied [18], [25], [26], [19] including the spanned geographic distances, routing policies, etc.

Delay-based IP geolocation includes a broad class of techniques aiming to calculate the geographic location of a client based on the delays observed between the client and a set of landmarks with known locations [27]. Most techniques apply regression analysis to find a function that best models the relationship between the measured delays and geographic distances [12], [17]. Multilateration is then used on the distances mapped between the landmarks and the client to constrain the region where the client is located. Recent techniques incur a median error of as low as a few kilometres [12]. To infer distances from delays, the speed at which packets are transmitted over the Internet has been approximated to 4/9 the speed of light in vacuum, a ratio called the Speed of the Internet (SOI) [28]. However, the actual speed is affected by several factors such as time of the day, region and characteristics of the underlying network. Based on 19 million RTT measurements in the Internet, Landa *et al.* [19] found that the knowledge of the geographic distance between two nodes, their /8 IP prefixes, and their countries can help scope down delay-estimation errors to within $\sim 22ms$.

Network Coordinates Systems (NCSs) [29] model a network as a geometric space by assigning coordinates to each node in the network. The coordinates denote a node's position relative to other nodes in the *network delay space*, i.e., according to its delay to/from them. One essential advantage of NCSs is the ability to locate a node's network position relative to *almost all* other nodes without overwhelming the network with storms of delay sampling [30]. NCSs are vulnerable to an adversary falsifying its coordinates [31].

The aforementioned delay studies provide solid evidence of a strong correlation between Internet delays and geographic distances [32], which is commonly speculated to stem from improved global network connectivity [27]. CPV leverages these results to address location verification.

3 Threat Model

The adversary is a human user that programs its client software to evade a geolocation process, to intentionally misrepresent its location. The adversary is in physical possession of the client device (e.g., laptop or smartphone), which is connected to the Internet and thereby to the LSP. The adversary has full control over its client device; it can install/uninstall any software.

We consider within scope an adversary that uses public proxies, Virtual Private Networks (VPNs) and/or anonymizers to hide its IP address or to hide any other identifying information that may reveal its true location. The adversary is also capable of manipulating delays [14].

CPV is designed to verify the output of a geolocation technique. The adversary must thus be able to mislead that technique first to forge its location. We assume, for simplicity, that the geolocation step prior to the operation of CPV is an unverified location assertion; CPV is then to verify this assertion. By considering this case, whereby the adversary can simply assert a location (e.g., the LSP asks its users to simply input their location), the adversary is powerful enough to evade any basic geolocation technique.[3]

We define the *target location* as the location the adversary attempts to appear at. The following two use cases explain

2. The accuracy of this protocol is analyzed in a separate paper [21].

3. Some geolocation techniques are harder to evade than others [14].

adversarial motivation to forge location, both of which are within the threat model.

Impersonation. To mitigate online impersonation of users' accounts, typically done through password-guessing attacks, logins can be restricted to location(s) (e.g., country) associated with the legitimate user's account. To impersonate a user, the adversary needs to not only guess the user's password, but also the user's associated location, and place itself fraudulently in that location. In this case, the adversary's target location changes arbitrarily according to the account being attacked.

Violation of geographic-restriction policies. When an LSP customizes its services/content based on the location of its users, such as location-sensitive multimedia providers (e.g., Hulu and Pandora), adversaries may be motivated to evade geolocation to gain location-dependent benefits. This threat is harder to defend against than the previous one, since the adversary's target location is fixed, and immediately known to the adversary.

4 CPV: Client Presence Verification

CPV builds on the established result that Internet delays and geographic distances have strong positive correlation [33] (see §2). In CPV, when a client asserts its presence in a geographic location, delays are measured between the client and three *verifiers*[4] encompassing the asserted location. These delays are then processed to provide assurance that the client is truly present (geographically) inside the triangle determined by the three verifiers. The size of that triangle is the verification granularity; as detailed in §6 below, experimental results show that CPV can provide assurance granularity down to a triangle whose area is as small as a circle of radius $\sim 400km$.

To reduce falsely rejecting legitimate (honest) clients and falsely accepting adversaries, factors affecting the delay-distance correlation (e.g., route circuitousness, queuing delays and congestion) must be addressed. The forward and reverse paths between any two hosts over the Internet are often affected by those factors differently, resulting in delay asymmetry [20]. The less affected path is likely to be the faster one (i.e., with a smaller one-way delay), and thus better represents the distance between the two hosts. Relying on the smaller one-way delay (OWD) between the client and the verifiers rather than the round-trip times is, thus, expected to improve CPV's accuracy in judging location assertions.

Classical OWD-estimation protocols (e.g., OWAMP [24]) can be exploited by dishonest clients as they typically require heavy client cooperation—the honest client synchronizes its clock with the server, calculates and reports its view of the delays. Such techniques cannot be used by CPV because client dishonesty is assumed. A commonly used method to estimate OWDs when cooperation cannot be achieved is to estimate the RTT, and halve it. This method is less accurate than OWAMP-like protocols [34]. We introduce a novel OWD-estimation protocol, which can be more accurate than halving the RTT [21], yet requires less cooperation than required by OWAMP-like protocols. This protocol is detailed in §5 below. Such accurate OWD-estimation is one measure utilized by CPV for accurate delay-to-distance mapping. By the end of this section, a summary is provided on how CPV manages the delay-measurement process to reduce the factors affecting this mapping, without jeopardizing the integrity of the location verification process.

4. In practice, verifiers could be dedicated servers maintained by an independent party providing location verification as a service.

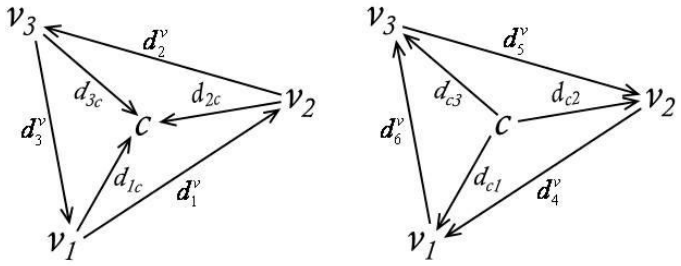


Fig. 1. Notation of OWDs between client c and verifiers v_1, v_2 and v_3.

After mitigating these factors, CPV uses a simple function to map delays to distances, and verifies assertions based on these distances (see §4.3 below).

4.1 Operational requirements

CPV requires geographically-distributed verifiers whose locations are consistent with the LSP's *Permitted Geographic Regions* (PGRs). PGRs are the regions in which clients are permitted to receive services/content or carry out location-specific operation (e.g., login or vote). The client must not control any of the verifiers involved in corroborating its assertion. To successfully enforce the LSP's location-aware policies, the verifiers must:

1) be publicly reachable over the Internet;
2) have a public-private key pair, with each verifier aware of the public keys of all other verifiers in the set, possibly through a closed Public Key Infrastructure (PKI); and
3) the convex hull of the verifiers must encapsulate the LSP's PGR.

4.2 Notation and definitions

The set of verifiers available to the LSP is denoted $\mathbb{V}$. For any triangle, $\triangle$, the set of the three verifiers determining $\triangle$ is denoted $V_\triangle \subset \mathbb{V}$. For any geographic location $l = \{\text{latitude, longitude}\}$, E_l is the set of triangles enclosing l, such that all $\triangle_l \in E_l$ are near equilateral in the network delay-space (see §2), and do not cross the PGR border.

There are three bidirectional edges joining a client with three verifiers, and three bidirectional edges joining the three verifiers, as shown in Fig. 1. Each of the six edges has two OWDs in opposite directions. Denote $\mathbf{D}^\bullet$ as an ordered list holding six OWD estimates at a given time. The estimates correspond to the smaller of the forward and reverse OWDs (i.e., at current network conditions) at each of the six bidirectional edges in Fig. 1. The superscript $\bullet$ is the protocol used to estimate the delays in $\mathbf{D}^\bullet$.

A client and three verifiers make four triangles. The function $valid(\mathbf{D})$ checks for triangular inequality violations (TIVs) in the four triangles whose side lengths are mapped from the six OWDs in $\mathbf{D}$. It returns true only if, for each of the four triangles, the sum of each two sides is greater than the third. The function $area\_v(\mathbf{D})$ calculates the area of the triangle determined by the three verifiers; the side lengths of that triangle are mapped from the three OWDs in $\mathbf{D}$ that belong to the edges between the verifiers. The function $area\_c(\mathbf{D})$ similarly

calculates the areas of the three triangles determined by each pair of verifiers and the client, and returns the summation of those areas.

4.3 CPV description

CPV's verification process begins with an asserted client location as input, $l = \{lat, lon\}$. The LSP chooses a triangle $\triangle_l \in E_l$, and informs the client of the IP addresses of the verifiers in $V_{\triangle_l}$. The client connects to the verifiers[5] and the verification process, Algorithm 1, begins.

Algorithm 1: Executed by the verifiers in $V_{\triangle_l}$ when a client asserting to be at location l connects to them.

Input: Number of iterations, $n_{\triangle_l}$; tolerance of area inequality, $\epsilon_{\triangle_l}$; and acceptance threshold $\tau_{\triangle_l}$
Output: Accept/Reject client's location assertion
begin

1 $pass := 0$
2 **for** $i := 1$ **to** $n_{\triangle_l}$ **do**
3 $\mathbf{D}_i := \phi$
4 Estimate, in real time, the one-way delays for $\mathbf{D}^{mp}$ and $\mathbf{D}^{av}$ using Algorithm 2 (see §5).
5 **if** *valid*($\mathbf{D}^{mp}$) **then** $\mathbf{D}_i := \mathbf{D}^{mp}$
6 **else if** *valid*($\mathbf{D}^{av}$) **then** $\mathbf{D}_i := \mathbf{D}^{av}$
7 $\delta_i := area\_c(\mathbf{D}_i) - area\_v(\mathbf{D}_i)$
8 **if** $\mathbf{D}_i \neq \phi$ **and** $\delta_i \leq \epsilon_{\triangle_l}$ **and** $acceptable(\mathbf{D}_i)$ **then**
9 $pass := pass + 1$
10 $\Gamma := pass / n_{\triangle_l}$
11 **if** $\Gamma < \tau_{\triangle_l}$ **then**
12 Reject client's location assertion
13 **else**
14 Accept client's location assertion

First (in line 4), the verifiers estimate the *smaller* of the forward and reverse OWDs at the six edges between the verifiers and the client (see Fig. 1) using two protocols: *Minimum Pairs* (mp) and *Average* (av), as explained in §5. The six OWDs are then mapped to distances according to the simple mapping function $f(x) = x$, i.e., x ms is equal to x km. The resulting distances are never used in an absolute form; they are only processed relative to each other. This design provides the advantage of resilience to factors that affect the network comprising the client and the three verifiers, e.g., a network congestion that affects the delays of the six edges altogether.

OWD estimation is done iteratively (line 2), where the input parameter $n_{\triangle_l}$ specifies the number of iterations to be performed, to account for possible delay instability [37]. The *confidence ratio*, Γ (line 10), represents the verifiers' confidence of the truthfulness of the asserted location. It is calculated as the proportion of iterations where the values of $area\_c(\mathbf{D}^{mp})$ and $area\_v(\mathbf{D}^{mp})$ (see §4.2) match within a suitable error tolerance, $\epsilon_{\triangle_l}$. From a geometric perspective, we have the following claim (see the appendix for proofs):

Claim 1. Let P be a point in the Cartesian plane, and let $\triangle XYZ$ be the triangle determined by the points X, Y and Z. If P is strictly outside $\triangle XYZ$, then the sum of the areas of $\triangle XYP$, $\triangle XPZ$ and $\triangle PYZ$ is greater than the area of $\triangle XYZ$.

5. The client may use websockets [35] to connect to the verifiers, as a stable means of delay measurement through the browser [36].

TIVs are evident in the Internet [38]. Because CPV relies on triangular areas in verifying location assertions, TIVs can thwart CPV's successful operation. Additionally, an adversary can manipulate the OWD-estimation process (§5), overly flattening some triangles and (artificially) resulting in TIVs. Thus, the verifiers become less confident about the truthfulness of the asserted location as more TIVs occur, which is a security precaution to reduce potential false accepts. This can be seen in line 8, where $\mathbf{D}_i$ must hold a valid set of delays (from lines 5 or 6) for Γ (line 10) to increase.

Iterating the delay-estimation process helps reduce the number of benign TIVs [25], hence reducing the number of false rejects. Additionally, more than one delay-estimation protocol (namely, both mp and av) further lessens the effect of TIVs. In lines 5 and 6, $\mathbf{D}^{mp}$ is checked first because it is more resilient to delay spikes, as discussed in §5 below.

The error tolerance, $\epsilon_{\triangle_l}$ (line 8), accounts for route circuitousness [26], congested routes, or other factors that contribute to inaccuracies in the delay-distance mapping over the Internet. If an adversary's true location is far from the asserted location that one of the *inner* triangles (those having the client as one of their vertices) becomes obtuse, the triangle becomes flattened and its area decreases. An unnecessarily large error tolerance may thus falsely accept this adversary.

To mitigate this effect, we include the $acceptable(\mathbf{D})$ function (line 8), which checks that the OWD between verifier v and the client is not larger than the OWDs between v and the other two verifiers. The function returns true only if the previous statement is true for the three delay-mapped distances in $\mathbf{D}$ that are between the client and the verifiers. From a geometric perspective, using the notation $\overline{AB}$ for the length of line segment AB, we have the following claim (see the appendix for proofs):

Claim 2. Let W be a point in the Cartesian plane, and let $\triangle XYZ$ be the triangle determined by the points X, Y and Z such that $\overline{XZ} \leq \overline{XY}$. If $\overline{XW} > \overline{XY}$, then W is strictly outside of $\triangle XYZ$.

Calibration of input parameters. To set the three input parameters of Algorithm 1 for each $\triangle$, the three verifiers in $V_{\triangle}$ can operate CPV to verify the geographic presence/absence of network nodes that are known (as a ground-truth) to be inside/outside $\triangle$ (e.g, using other verifiers in $\mathbb{V}$). Based on the delays between the verifiers and these nodes, the input parameters should be set such that CPV accepts inside nodes, and rejects outside ones. For example, in line 10 (Algorithm 1), if $\Gamma \geq 0.6$ for all such nodes, then $\tau_{\triangle_l}$ should be set to 0.6.

Summary. CPV's measures to reduce factors affecting delay-to-distance mapping can be summarized as follows:

1) Two protocols are used to estimate OWDs instead of one to reduce the effect of TIVs.
2) Active delay measurement is used with each client, which reflects the most recent delay status in the region [37].

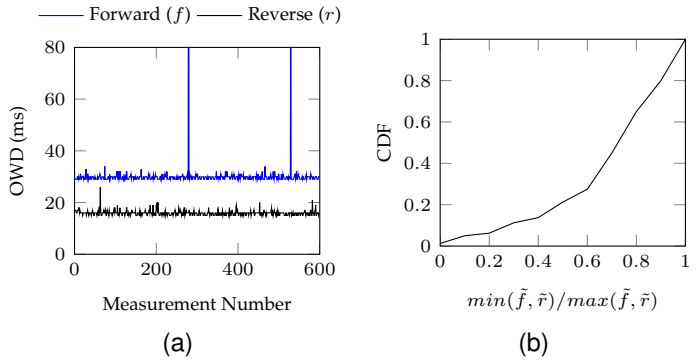


Fig. 2. (a) An example of 600 forward and reverse OWD measurements between two verifiers that are $\sim$950km apart; (b) The ratio $min(\tilde{f},\tilde{r})/max(\tilde{f},\tilde{r})$ between 80 pairs of verifiers, where $\tilde{f}$ and $\tilde{r}$ are the medians of the 600 forward and reverse OWD measurements respectively. A point (x, y) on the curve of (b) means the proportion y of the 80 pairs had a ratio $\leq x$.

3) No universal delay-to-distance mapping is used. Rather, mapping is done relative to other delays in the region.
4) Delay-estimation is conducted iteratively to more accurately converge to the actual delays at current network conditions [39].
5) The three verifiers are chosen within a geographical proximity of the asserted location to
   a) reflect regional delays [18], [17];
   b) span fewer Autonomous Systems, which reduces route circuitousness [33];
   c) reduce the number of TIVs [25]; and
   d) exhibit stronger positive correlation between delays and distances [19].

5 One-Way Delay Estimation

Delays over the Internet are asymmetric [20]. For example, Fig. 2a plots 600 forward (f) and reverse (r) OWDs measured between two verifiers in our experiments (explained later in §6) over the course of an hour using an OWAMP-similar approach. Their medians are $\tilde{f}$ = 30ms and $\tilde{r}$ = 16ms respectively. The ratio $min(\tilde{f},\tilde{r})/max(\tilde{f},\tilde{r})$ is $\sim$0.5.

Figure 2b shows a Cumulative Distribution Function (CDF) of $min(\tilde{f},\tilde{r})/max(\tilde{f},\tilde{r})$ between 80 pairs of verifiers in our experiments. The smaller of $\tilde{f}$ and $\tilde{r}$ was less than half the larger in 13% of the cases, and less than three-quarters the larger in 55% of the cases. These numbers highlight the asymmetry of delays in Internet routes, as established in the literature [20].

5.1 Adversarial model while estimating OWDs

To measure OWDs, the verifiers cannot trust the potentially dishonest client to synchronize its clock with theirs and exchange timestamps. While estimating OWDs, the verifiers must assume the client may

- drop/reject timestamp messages;
- refrain from appropriately synchronizing its clock with the verifiers;
- submit a forged OWD; or
- falsify the timestamp before reporting it.

To account for these threats, a protocol called the *Minimum Pairs* (mp) is introduced to estimate OWDs.

5.2 Clock synchronization among the verifiers

In mp, the verifiers may choose to synchronize their clocks to the nearest ms to increase the accuracy of OWD estimates [40], [41], or use techniques that do not require accurate synchronization [42]. For example, Gurewitz *et al.* [43] proposed a technique that estimates OWDs in the absence of accurate clock synchronization between network nodes. Strong cooperation between these nodes is, however, required. The nodes conduct many OWD measurements among themselves using the poorly synchronized clock, and use those preliminary estimates to derive constraints of an objective function. The function uses optimization techniques, and reaches a per-link OWD estimate that minimizes the error with respect to the provided constraints.

While this class of techniques addresses imperfect clock synchronization, the mp protocol addresses client untrustworthiness. Therefore, such a class of techniques can be used among the verifiers if accurate clock synchronization cannot be achieved. However, due to its strong cooperation and trustworthiness requirement, it cannot be used with potentially dishonest clients.

5.3 The Minimum Pairs protocol *(mp)*

Algorithm 2 details the mp protocol. Notation:

- $S_a(m)$ denotes message m digitally signed by entity a.
- $A \xrightarrow{m} B$ means A sends message m to B.
- t_a is the most recent timestamp according to verifier a's clock.
- The variable e_{ij} in line 10 corresponds to $d_{ic} + d_{cj}$ (see Fig. 1).

The three verifiers take turns to send the client digitally signed timestamps of their most recent system time (line 3). Once received, the client is required to forward this message to the three verifiers.[6] Failing to promptly forward the message, or tampering with the timestamp, leads to the rejection of the client's assertion, as discussed later in §7.

When all three verifiers are done their turns, they will have nine values of delays corresponding to $d_{ic} + d_{cj}$ for all $1 \leq i, j \leq 3$. The mp protocol estimates the smaller of d_{ic} and d_{ci} independently, for all $1 \leq i \leq 3$, as follows. First, for all $1 \leq i, j \leq 3$ and $i \neq j$, the larger of $d_{ic} + d_{cj}$ and $d_{jc} + d_{ci}$ are discarded (line 13) because the smaller sums are likely to correspond to the smaller OWDs. Second, the three remaining sums are equated to the corresponding smaller OWDs, and estimates to the smaller delays are obtained by solving simultaneously for x_1, x_2, x_3:

$$x_i + x_j = min(d_{ic} + d_{cj},\ d_{jc} + d_{ci}) \quad \forall\ 1 \leq i < j \leq 3$$

where x_i is the estimate to the smaller of d_{ic} and d_{ci}. Working out the three equations is demonstrated in lines 13 to 17 of Algorithm 2.

Discarding the larger delays (line 13) provides a fundamental advantage to mp over av, as it helps reduce the unfavourable effect of delay spikes occurring in one direction but not the other. Compared to av, the probability of mp to exclude delay spikes is higher [21]; thus, the mp protocol is given priority over av in Algorithm 1.

In line 12, estimating the smaller OWDs of the edges between the verifiers (i.e., d_i^v in Fig. 1) is simpler, since the

6. This behavior can be implemented in the browser through javascript.

Algorithm 2: The mp algorithm. See notation inline.

Input: The set of the three verifiers, V (see Fig. 1).
Output: $\mathbf{D}^{mp}$ and $\mathbf{D}^{av}$
begin

1 **foreach** v_i **in** V **do**
2 v_i retrieves its current system time $b := t_i$
3 $v_i \xrightarrow{b, S_i(b)}$ client
4 **foreach** v_j **in** V **do**
5 client $\xrightarrow{b, S_i(b)}$ v_j
6 v_j records the message-receiving time $r := t_j$
7 v_j validates $S_i(b)$
8 **if** *invalid signature* **then**
9 Abort "possible client cheating attempt"
10 $e_{ij} := r - b$
11 **for** $i := 1$ **to** 6 **do**
12 The verifiers in V measure d_i^v (see Fig. 1)

/* Calculating $\mathbf{D}^{mp}$ */
13 $m := \{min(e_{12},\ e_{21}),\ min(e_{23},\ e_{32}),\ min(e_{31},\ e_{13})\}$
14 **for** $i := 1$ **to** 3 **do**
15 $j = ((i + 1) \bmod 3) + 1$
16 $k = (i \bmod 3) + 1$
17 $x_i := (m_i + m_j - m_k)/2$
18 $y_i := min(d_i^v, d_{i+3}^v)$
19 Append x_i and y_i to $\mathbf{D}^{mp}$

/* Calculating $\mathbf{D}^{av}$ */
20 **for** $i := 1$ **to** 3 **do**
21 $x_i := e_{ii}/2$
22 $y_i := (d_i^v + d_{i+3}^v)/2$
23 Append x_i and y_i to $\mathbf{D}^{av}$

24 **return** $\mathbf{D}^{mp}$ and $\mathbf{D}^{av}$

verifiers trust each other; for example, the OWAMP [24] tool can be used. Again, the verifiers discard the larger of the forward and reverse OWDs for each of the three edges between them (line 18). Finally, the set $\mathbf{D}^{mp}$ holds the six smaller OWD estimates (line 19).

Using av as well. In lines 20 through 23, the verifiers also use the *average* (av) protocol, to obtain $\mathbf{D}^{av}$, which is used in Algorithm 1 (§4) as a fallback if the estimates in $\mathbf{D}^{mp}$ result in TIVs [25]. The av protocol simply calculates half the RTTs [44] (lines 21 and 22).

6 Evaluation

To evaluate CPV, we use the rates of *false rejects* (FRs) and *false accepts* (FAs) as the assessment metrics. If a client asserts to be at l, an RF occurs when this client is actually present somewhere inside $\triangle_l$, and is judged by the verifiers in $V_{\triangle_l}$ as absent from $\triangle_l$. By contrast, an FA occurs when that client is actually absent from $\triangle_l$, and is judged by the verifiers in $V_{\triangle_l}$ as present in $\triangle_l$.

We used 80 PlanetLab [23] nodes in USA and Canada (Fig. 3), and identified 34 different sized triangles satisfying the requirements stated in §4. The triangles were chosen with internal angles ranging 50-70 degrees so as to be near-equilateral in the network delay-space, as specified in §4.2. Triangular areas

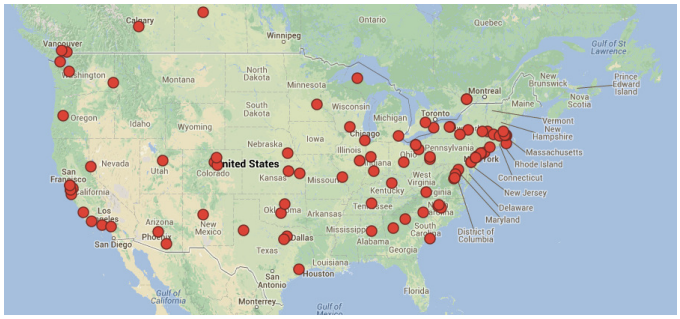


Fig. 3. Locations of the 80 PlanetLab nodes used in our experiments. Map data: Google, INEGI.

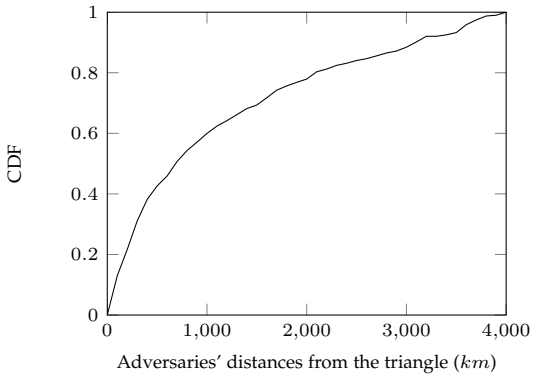


Fig. 4. Adversaries' distances from the triangles' closest side. A point (x, y) means the proportion y of adversaries were $\leq x$ km away from the closest side. Note: this graph shows experimental design, not results.

ranged from ∼32,000km^2, almost the size of Maryland state, to ∼500,000km^2 (or a circle of radius ∼400km), almost the size of Spain.

We assumed that the PGR is a triangular-shaped region that perfectly coincides with the dimensions of the triangle. One triangle was considered at a time. For each triangle, all nodes—except the three determining the triangle—acted as clients; all clients asserted to be at the centroid of that triangle. Combining clients of all triangles, *legitimates* (clients actually inside) totalled 146 and *adversaries* (clients actually outside) totalled 2,301 for a total of 2,447 experiments. The verifiers determining each triangle were verifying assertions of all clients concurrently. The verifiers used the network time protocol (NTP) [45] to synchronize their clocks. Knowing the ground truth of legitimates and adversaries with respect to each triangle, our objective is to identify the optimal values of $\epsilon_\triangle$ and $\tau_\triangle$ for each of the 34 triangles, and quantify the FRs and FAs at these values.

Figure 4 shows a CDF of the adversaries' distances from the triangles' closest side. Half the adversaries were less than 700km away from the triangle's closest side (i.e., the triangle encapsulating their fraudulently asserted location), and no adversary was farther than 4,000km away. For reference, the width of the United States is approximately 4,000km. The argument is that if CPV rejects relatively nearby adversaries, it will reject more distant ones.

Experiments were run over the course of a month (April 2013) and at different times of the day. The number of iterations, $n_\triangle$ (Algorithm 1), was fixed at $n_\triangle = 600$ for all $\triangle$ in the 34-triangle set to study the factors affecting CPV over a

TABLE 1
Results for clients D, E, and F. The "§" column shows the section where each variable (row) is analyzed further.

Variable	Client D	Client E	Client F	§
Number of TIVs	114	11	0	6.2
$\tilde{\delta}$ (km^2)	30	66	209	6.3
Γ (0 to 1)	0.84	0.2	0	6.4

TIV = Triangular Inequality Violation;
$\tilde{\delta}$ = median of area differences; Γ = confidence ratio.

relatively long period of time (a total of ∼13.3 million delay measurements were taken between all nodes). Fewer iterations might be sufficient to judge a client, as we show in §6.6 below.

6.1 An example

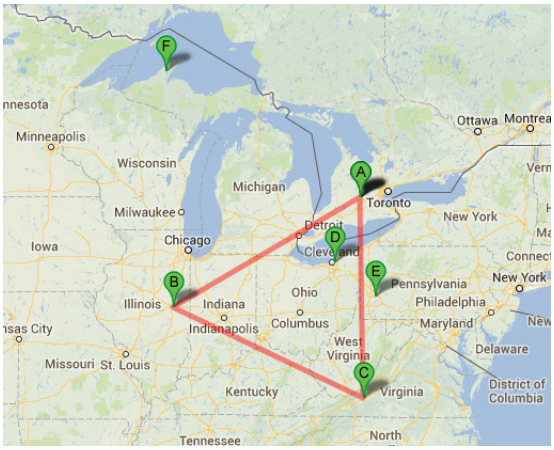
We detail the results of one of the triangles in our 34-triangle set, and three of the clients being verified by that triangle. One of the clients was legitimate, the other two were adversaries. Figure 5a shows the geographic location of the triangle and the three clients, labelled D, E and F. The area difference, δ_i (line 7 of Algorithm 1) for all $1 \leq i \leq 600$, is plotted for the three clients in Fig. 5b.

Number of TIVs. Some iterations have no corresponding values for the area difference (visible in high resolution). Those are the ones where $valid(\mathbf{D}^{mp})$ and $valid(\mathbf{D}^{av})$ (lines 5 and 6 of Algorithm 1) returned false, i.e., the mapped distances resulted in at least one TIV of the four triangles determined by the three verifiers and the client. Of all 600 iterations, the number of iterations where both functions returned false for D, E and F are 114, 11 and 0 respectively.

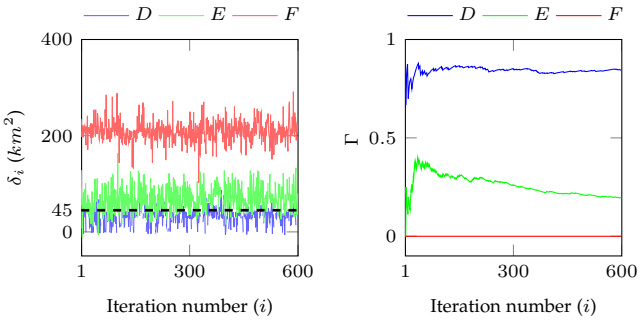
Area difference (δ). From Fig. 5b, the median of δ_i, $\tilde{\delta}$, for clients D, E and F is $30km^2$, $66km^2$ and $209km^2$ respectively. The median corresponding to F is substantially larger than that of D and E because F is relatively far away from the triangle. The smallest recorded area difference for F is $\delta_{325} = 102km^2$. Therefore, any value for $\epsilon_\triangle$ in the range $\epsilon_\triangle < 102$ keeps the variable $pass = 0$ (line 9, Algorithm 1) for all iterations, resulting in $\Gamma = 0$. Consequently, at $\epsilon_\triangle < 102$, any value for $\tau_\triangle$ (the acceptance threshold, §4) in the range $\tau_\triangle > 0$ rejects F's assertion. Client E was less than $50km$ away from the triangle's nearest side AC, thus the average area difference of E is close to that of D. However, at $\epsilon_\triangle = 45$, there is a visible distinction between both nodes—there existed a value for $\epsilon_\triangle$ (i.e., $45km^2$) that enabled the verifiers to correctly judge the assertions of both clients, D and E, despite being geographically collocated.

Confidence ratio (Γ). In Algorithm 1, Γ is calculated when all n iterations are performed. Figure 5c plots Γ (at $\epsilon_\triangle = 45km^2$), assuming it was calculated at each iteration. Despite the relatively close values of δ_i between D and E in Fig. 5b, their Γ greatly differs. At $i = 100$, Γ is 0.86 and 0.3 for D and E respectively. Therefore, after 100 iterations, any $\tau_\triangle$ in the range $0.3 < \tau_\triangle \leq 0.86$ enables the verifiers to decide that D is a legitimate and E is an adversary. When all 600 iterations are performed, Γ becomes 0.84 and 0.2 for D and E respectively, showing no significant change from the 100^{th} iteration.

Summary. Table 1 summarizes the results of this example. The following three subsections analyze each of the three variables (rows) in the table for all 2,447 experiments. The respective subsection is reported in the table.



(a) The area of the shown triangle is ∼230,000km^2. Clients E and F are outside, whereas D is inside. Map data: Google, INEGI.



(b) Area differences (c) At $\epsilon_\triangle$ = 45km^2

Fig. 5. An example from our experiments showing a triangle and three clients (best viewed in color).

6.2 Triangle inequality violation (TIV)

For each client, four delay-based triangles are calculated at each iteration, three of which have the client as one of the triangle's vertices for a total of 3×600 = 1,800 triangles involving the client. Figure 6 shows a CDF of the number of TIVs, resulting from either mp- or av-estimated delays, for each client (legitimate or adversary). Note that Algorithm 1 does not call the function $valid(\mathbf{D}^{av})$ if $valid(\mathbf{D}^{mp})$ returns *true* (line 5). We thus counted the number of TIVs for av independent of the algorithm operation.

For the triangles described by mp-estimated delays, very few clients (5%) suffered no TIVs, and 86% suffered at least 10 (of 1,800 possible) TIVs. While these results confirm that TIVs are evident in the Internet [46], they emphasize the importance of iterative delay-measurement to mitigate TIVs. For example, half the clients suffered fewer than 28% (or 500) TIVs in total, enabling CPV to use the remaining 1,300 valid triangles to verify location assertions.

The case was slightly different using av-estimated delays; almost all clients suffered at least one TIV and 93% suffered at least 10 of the possible 1,800 TIVs. However, av was overall better in avoiding TIVs than mp. Half the clients suffered fewer than 300 TIVs (versus 500 for mp). Because av estimates the OWD of a triangle's side as the average of both directions, it tends to reduce the discrepancy between the three sides, leading to fewer TIVs than mp.

The number of TIVs resulting from the delays measured

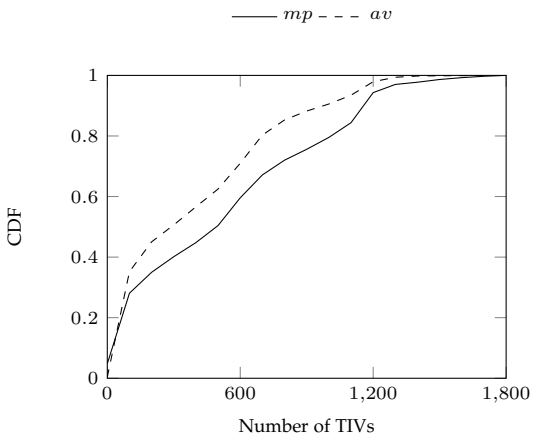


Fig. 6. Number of TIVs involving the client. A point (x, y) means the proportion y of clients suffered x or fewer TIVs.

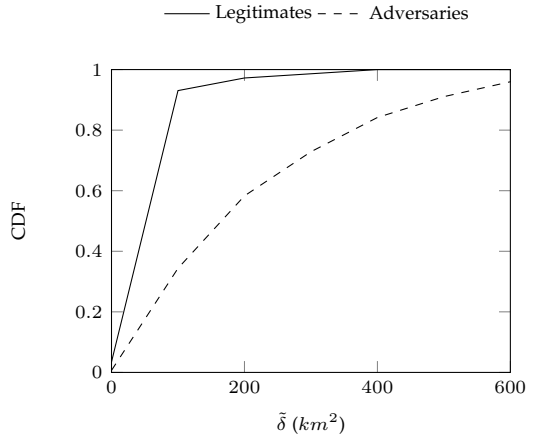


Fig. 7. Median area difference ($\tilde{\delta}$) for 146 legitimates, and 2,301 adversaries. A point (x, y) means $\tilde{\delta}$ was $\leq x\ km^2$ for the proportion y of clients.

by each protocol highlights the importance of leveraging both to backup each other, as CPV operates (see Algorithm 1). The benefit that CPV gains when relying on both is further analyzed in §6.7.

6.3 Triangular area as a discrimination metric

We analyze the effectiveness of using triangular areas as a metric to distinguish legitimates from adversaries. Figure 7 shows a CDF of the median area difference, $\tilde{\delta}$, for all 146 legitimates and 2,301 adversaries. These area differences are either calculated from the mp or the av protocols (see Algorithm 1). Note that, from Fig. 4 (§6), about one-third of all adversaries were within $400km$ of the triangle's sides (e.g., E and F in Fig. 5a were within $50km$ and $850km$ of the triangle's side respectively).

The results in Fig. 7 show that 93% of all legitimates had $\tilde{\delta}$ $<100km^2$, whereas two-thirds of all adversaries had more than that value. The results affirm that, although the experiments involved numerous adversaries that are close to the sides of the triangles encompassing their asserted location, *triangular areas* distinguished between them. In conclusion, the triangular area served as a successful discrimination metric to distinguish between legitimates and adversaries.

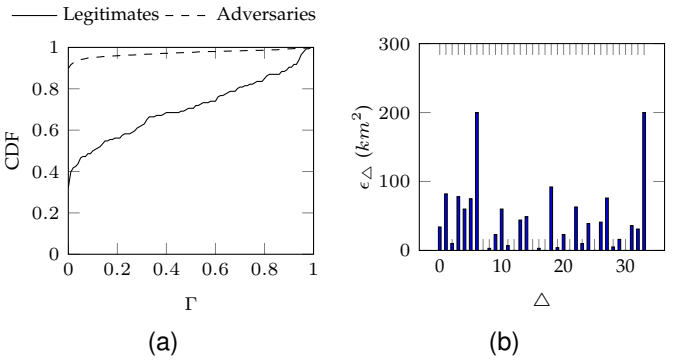


Fig. 8. (a) Confidence ratios (Γ) for 146 legitimates, and 2,301 adversaries. A point (x, y) means Γ was $\leq x$ for the proportion y of clients. (b) $\epsilon_\triangle$ for each $\triangle$ in all 34-triangles.

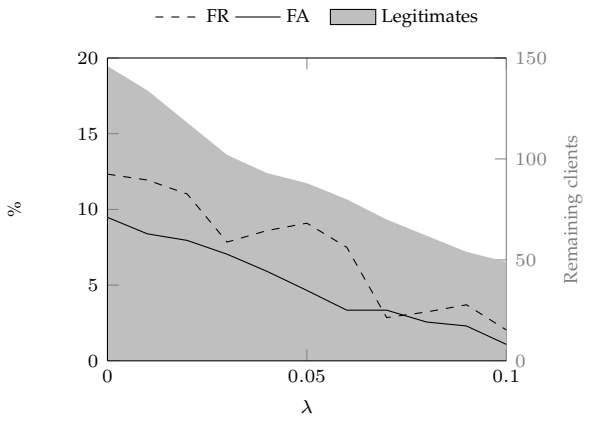


Fig. 9. FRs and FAs when legitimates at location $g = \{x, y\}$ are excluded from the experiments, such that $away(\triangle, g) < \lambda$. The shaded region is the number of remaining legitimates.

6.4 The confidence ratio (Γ)

Figure 8a shows the CDF of Γ for legitimates and adversaries; the values of Γ associated with 90% of all adversaries was 0, i.e., the verifiers were confident about the absence of those adversaries from the triangles encompassing their asserted location. The case was different with legitimates, where only 30% had a Γ value above 0.5, and half had a value above 0.1. Thus in our experiments, CPV detected falsified location assertions easier than realizing the correctness of true (honest) assertions. The values of $\epsilon_\triangle$ that result in this Γ distribution are shown in Fig. 8b.

6.5 Proximity to triangle's sides

This subsection analyzes the effect of a legitimate's proximity to the sides of its enclosing triangle. Let $away(\triangle, g)$ be the ratio of the distance between a point g inside $\triangle$ and side $z^g_\triangle$ to the length of $z^g_\triangle$, where $z^g_\triangle$ is the closest side to g. If $away(\triangle, g) = 0$, then g lies on one of the three sides of $\triangle$. We evaluate CPV's improved efficacy as the number of legitimates close to the sides (i.e., with relatively small values of $away()$) decreases.

Figure 9 shows the number of FRs and FAs after excluding legitimates at locations g, such that $away(\triangle, g) < \lambda$ for all $0 \leq \lambda \leq 0.1$. The number of remaining legitimates is shown on the same chart as the y-axis on the righthand side.[7] All

7. Most of the used PlanetLab nodes are located within cities, which explains the relatively large number of nodes close to triangles' sides.

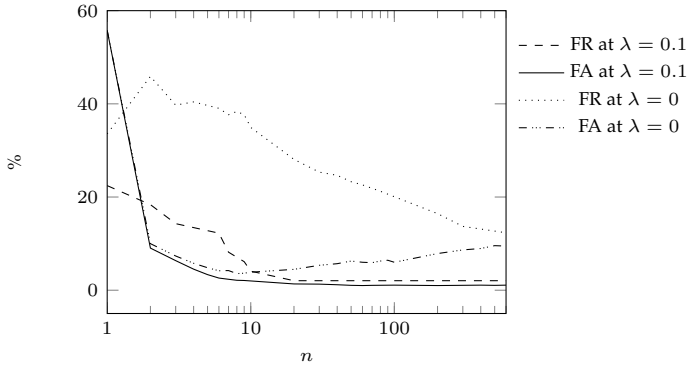


Fig. 10. FRs and FAs when n iterations are performed ($\log_{10}$ scale).

adversaries are included in the plot regardless of their triangle proximity. As more legitimates are excluded, the effect of the remaining ones on the FRs increases. When the remaining clients suffer relatively high network delays, the FRs oscillate as shown in the plot. Of the chosen PlanetLab nodes, we noticed three nodes suffering exceptionally high delays for unknown reasons. Their distance from the triangle's closest side was such that $0.002 \leq away() \leq 0.28$. Those nodes contribute to the oscillation intensity occurring in Fig. 9 as λ increases, and become very hard to partition from adversaries as more legitimates get excluded. At $\lambda = 0.1$, the FRs were 2% versus 12.3% at $\lambda = 0$. This improvement emphasizes the importance of appropriate triangle choice with respect to the asserted location. For an asserted location l, it is recommended that $\triangle_l$ be chosen such that $away(\triangle_l, l) \geq 0.1$.

Although the number of adversaries included in the experiments was unchanged over the spectrum of λ in Fig. 9, FAs improve as λ increases; the FAs were 9% at $\lambda = 0$, and dropped to 1.1% at $\lambda = 0.1$. Such improvement stems from the ability to find smaller ϵ values that do not falsely reject legitimates—now far from the triangle's sides, i.e., at $\lambda = 0.1$. Smaller ϵ values reduce FAs.

6.6 Number of CPV iterations

Figure 10 shows the change in FRs and FAs with the number of CPV iterations, n ($\log_{10}$ scale). FRs and FAs generally decrease as more iterations are performed, at $\lambda = 0.1$ and $\lambda = 0$. The results for $\lambda = 0.1$ are quite sensible: FRs and FAs decrease almost monotonically when more iterations are performed. With two iterations, at $\lambda = 0.1$, FAs dropped to $\sim$9% from over 50% when only one iteration was performed. Fewer than 10 iterations did not enable the verifiers to identify legitimates appropriately as the FRs were between 6% and 22%, i.e., no values for $\epsilon_\triangle$ and $\tau_\triangle$ existed to partition legitimates and adversaries. However, between 10 and 20 iterations, FRs and FAs, at $\lambda = 0.1$, levelled at $\sim$2% and $\sim$1% respectively.

At $\lambda = 0$, FAs dropped from $\sim$56% when one iteration was performed (at $n = 1$), to $\sim$10% when 9 iterations were performed (at $n = 9$). It then oscillated between $\sim$10% and $\sim$6% when fewer than 100 iterations are performed, climbing steadily to $\sim$8% for the rest of the iterations. This rise happened simultaneously with an improvement in the FRs (at $\lambda = 0$). As more iterations are performed, it becomes more feasible to find $\epsilon_\triangle$ values that partition legitimates from adversaries. To accommodate legitimates that are very close to the triangles' sides, large values of $\epsilon_\triangle$ were required, which resulted in falsely accepting more adversaries. This explains the rise in FAs as more iterations were performed, at $\lambda = 0$. Over the entire range of n, the FRs at $\lambda = 0$ decreased from $\sim$34% at $n = 1$ to $\sim$12% at $n = 600$. Even when legitimates are highly adjacent to their enclosing triangles' sides, large number of iterations can improve the ability of finding ϵ and τ values that better partition legitimates from adversaries. This highlights the importance of *iterative* delay-sampling, especially when the chosen verifiers determine a triangle whose sides are close to the asserted location.

6.7 Benefits of combining the *mp* and the *av* protocols

Table 2 summarizes PlanetLab results of different CPV evaluation scenarios. The columns represent modified versions of CPV, i.e., different from the behavior given in Algorithm 1. In line 4 of Algorithm 1, two OWD-estimation protocols are used (mp and av) to alleviate the effect of TIVs. Table 2 lists the results when only the av protocol is used ("av only" column), when only mp is used ("mp only" column), and when both are used ("CPV" column). The results are shown for various combinations of the exclusion threshold, λ (see §6.5), and the number of iterations, n. The table shows the FRs, the FAs, and their sum in each respective case.

From Table 2, the summation of FRs and FAs when both OWD-estimation protocols are used (right-most column under "CPV") is smaller in four out of six of the cases (table rows) compared to the summation when each protocol is used solely, e.g., 39 is less than 43 and 49 in the first case. Thus, the use of both OWD-estimation protocols tends to enhance the accuracy of the location verification process.

Using the mp protocol solely gave better results than av solely in four out of six cases. The av protocol was better at $\lambda = 0$ and $n \geq 100$. Recall from §6.2 that the mp protocol results in more TIVs. Since CPV counts the number of TIVs against the client, more TIVs tend to increase FRs, as shown by the results under the "mp only" column in Table 2. At $\lambda = 0$ and $n \geq 100$, there were 26% and 17% FRs using the mp protocol, versus 25 and 14% using av. In conclusion, CPV is best suited when utilizing both delay-estimation protocols to mitigate the unfavorable effect of TIVs.

7 Security Discussion

7.1 Classical geolocation attacks

Submitting false information. Although this may mislead simple geolocation techniques [5], it does not defeat CPV because the verification process (Algorithm 1) is independent of any information submitted by the client. §6 shows how CPV detects false location assertions (Fig. 11a) due to area mismatch or large client-verifier delays.

Using middleboxes. Some IP geolocation techniques can be circumvented if a client's IP address is concealed using generic *middleboxes* such as proxies, anonymizers, or VPNs [5]. These do not threaten the integrity of the verification process of CPV because delay measurements are conducted over the client's application layer. Middleboxes that blindly relay application-layer traffic (Fig. 11b) will also relay the timestamps (see §4) to the client [10]. A middlebox specifically designed to defeat CPV by searching application-layer traffic for timestamps could be mitigated using a proof-of-work mechanism [47].

TABLE 2
Benefits of using both, mp and av, for OWD-estimation in CPV. The table shows results of modified versions of CPV, where only one of the two protocols is used. The shaded column is the unmodified version, Algorithm 1.

Case	λ	n	av only			mp only			CPV (mp and av)		
			FR%	FA%	**FR+FA**	FR%	FA%	**FR+FA**	FR%	FA%	**FR+FA**
1	0	10	45	4.4	**49**	39	3.8	**43**	35	3.9	**39**
2	0	100	25	5.3	**30**	26	4.9	**31**	21	5.1	**26**
3	0	600	14	7.1	**21**	17	6.5	**24**	13	7.3	**20**
4	0.1	10	24	1.7	**26**	10	2.3	**12**	4.1	2.1	**6.2**
5	0.1	100	10	0.7	**11**	2.0	1.0	**3.0**	2.0	1.1	**3.1**
6	0.1	600	2.0	1.7	**3.7**	2.0	1.0	**3.0**	2.0	1.0	**3.0**

λ = legitimates exclusion threshold (see §6.5); n = number of iterations (see Algorithm 1); FR = false reject; FA = false accept; av is the Average protocol; mp is the Minimum Pairs protocol.

Manipulating delays to increase calculated distances. Delay-adding attacks [14] can be attempted on CPV when the adversary inserts a delay before forwarding timestamps. Assuming verifier i sent a timestamp, the adversary failing to forward it promptly to verifier j enlarges d_{ic} and d_{cj} fraudulently, increasing the value of $d_{ic} + d_{cj}$ (see Fig. 1 for notation). Because the mp protocol estimates the smaller OWD at each edge by solving simultaneous equations, selectively delaying timestamps can result in delay estimates that are smaller than the actual delay. For example, solving simultaneously the equations $a + b = 7$, $a + c = 8$ and $b + c = 9$ gives $a = 3$, $b = 4$, and $c = 5$. Whereas $a + b = 7$, $a + c = 8$, and $b + c = 13$ results in $a = 1$, $b = 6$ and $c = 7$. Thus, *increasing* $b + c$ resulted in a *smaller* value for a.

However, the adversary cannot reduce the *summation* of d_{ic} and d_{cj} as this requires speeding up the traffic propagation between the adversary and the verifiers [14]. From a geometric perspective, increasing the summation of any pair of edges does not help an adversary outside a triangle to forge its location making it inside. Formally, using the notation $\overline{AB}$ for the length of line segment AB, we have the following claim (see the Appendix for proofs):

Claim 3. Let P be a point in the Cartesian plane, and let $\triangle XYZ$ be the triangle determined by the points X, Y and Z. If P is strictly outside $\triangle XYZ$, then increasing the sums $\overline{XP} + \overline{PZ}$, $\overline{XP} + \overline{PY}$ or $\overline{YP} + \overline{PZ}$ without reducing at least one of the other sums cannot place P inside $\triangle XYZ$.

Manipulating delays to cause TIVs. As shown in Algorithm 1, CPV holds the number of TIVs against the client (the condition $\mathbf{D}_i \neq \phi$ in line 8 means $\mathbf{D}_i$ must not violate the triangle inequality to increment *pass*). In conclusion, manipulating delays does not help the adversary, but rather signals the adversary's evasion attempts.

7.2 Attempts to evade CPV

To study potential vulnerabilities in CPV, we review steps where the verifiers interact with the client.

Connecting to the verifiers. Assuming the adversary's target location (location it is trying to appear at) is l, connecting to a set of verifiers $V_{\triangle_{l'}} \neq V_{\triangle_l}$ does not help the adversary in pretending to be at l as those verifiers cannot verify the adversary's presence inside $\triangle_l$.

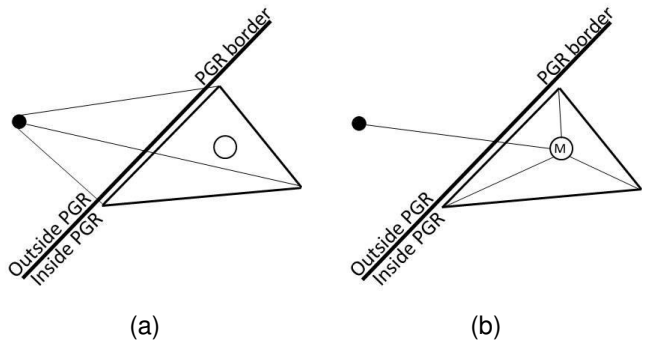


Fig. 11. An adversary asserting a false location (a) without using a middlebox, and (b) using a middlebox at the asserted location. •=true location; ○=asserted location; M=middlebox; PGR=Permitted Geographic Region.

Forwarding the timestamp. Because the verifiers sign the timestamps, the adversary can neither forge nor inject fake ones. Delaying a timestamp is discussed in §7.1.

7.3 Poor verifier deployment and PGR proximity

Adversaries bordering the PGR (Permitted Geographic Region) may be able to exploit inappropriate or insufficient verifier deployment. Figures 12a and 12b show examples of inappropriately deployed verifiers with respect to the PGR, where a triangle crosses the PGR border or encloses the PGR inside itself. As shown, a close adversary could be outside the PGR but inside those triangles. Verifying the presence inside the triangle does not ensure presence inside the PGR in those cases. Figure 12c shows potential vulnerability due to insufficient verifiers/triangles: not all regions inside the PGR are covered with triangles. The verifiers determining the shown (solid) triangle should not overly relax $\epsilon_\triangle$ to account for the uncovered region (relaxing $\epsilon_\triangle$ is depicted by the *dashed* triangle in Fig. 12c). Otherwise, the verifiers falsely accept an adversary close to the PGR asserting to be at the uncovered region of the PGR, as shown in Fig. 12c.

Possible countermeasures. To address PGR border crossing, additional overlapping triangles could be used to enclose the asserted location as long as a single triangle, or the intersection of multiple triangles, crosses the PGR border. The intersection region of the triangles must (1) not cross the PGR border and (2) enclose the asserted location, as shown in Fig. 13a. Client presence inside the PGR is then verified only if the verifiers of each triangle accept the assertion. For example, in Fig. 13a, if the client's (adversary's) true location was at any of the

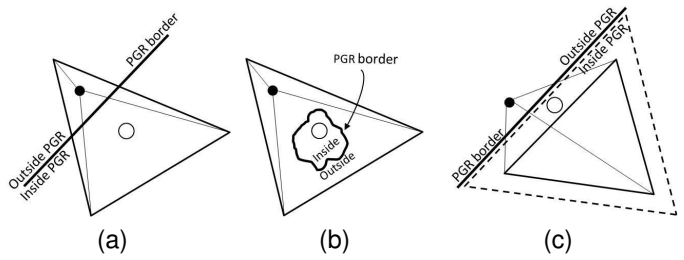


Fig. 12. (a) and (b) inappropriately deployed verifiers; (c) insufficiently deployed verifiers. •=true location; ○=asserted location; PGR=Permitted Geographic Region.

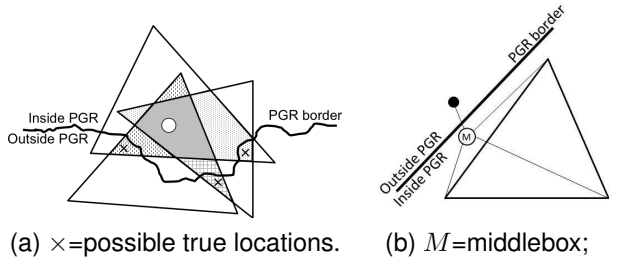


Fig. 13. Defences against a bordering adversary that exploits inappropriate or insufficient verifier deployment. •=true location; ○=asserted location; PGR=Permitted Geographic Region.

areas marked with $\times$, two triangles may falsely accept the assertion. Two triangles are insufficient in that case because the PGR border crosses the overlapping areas of each two of the three triangles. Verifying the presence inside all three suffices to verify the correctness of the assertion.

As for insufficient deployment of verifiers, whenever an assertion is made in a region not covered by any triangle, the LSP (location-sensitive provider) could use a measurement-based IP geolocation technique instead of relying on client-dependent geolocation (such as GPS). A bordering adversary must then evade this technique prior to bypassing CPV. It would then be challenging for the adversary to precisely target a location not covered by any triangle only through delay manipulation [14]. In such a case, using a measurement-based IP geolocation technique motivates the adversary to use a middlebox inside the uncovered region of the PGR (Fig. 13b). However, middleboxes tend to increase delays [10], which helps the verifiers detect the adversary's false assertion.

8 Related Work

Delay-based proximity verification has been well studied in contexts other than the Internet, such as single-hop wireless networks (e.g., Radio-Frequency Identifiers). Proposals include distance bounding protocols [48], ultra-sound based approaches [15], and region bounding through triangulation [16]. The nature of delays over the Internet differs from those in single-hop wireless networks. Internet delays alleviate some of the challenging problems in the single-hop wireless context (e.g., less sensitivity to processing delays), but introduce new challenges (e.g., stochastic queueing delays due to traffic/route uncertainty [17]). Thus, location verification over the Internet is a distinct research problem.

Privacy-aware location-proof architectures have been proposed to enable users to obtain proofs of their presence in a certain location, where an LSP-trusted access point is available [49], [50], [51]. A user gets the access point to assert the presence of their device within the access point's proximity. This is done by binding a secret and unique identifier of the user to the access point's location. These solutions assume users unwilling to disclose this identification credential. We do not make this assumption in our work and hence, these solutions target a different class of applications, where the user's motivation to preserve the confidentiality of their unique identification credentials exceeds their motivation to forge their location. If this is not the case, an adversary may—by the aid of a remote colluding party—send their identification credentials to get them bound to and endorsed by a remote access point, thus forging their location.

9 Conclusion

This paper leverages the strong delay-distance correlation established in the literature [27], [12], [17], [19], and presents CPV to verify geographic location assertions on the Internet. CPV is evaluated through detailed real world experiments on PlanetLab, and two main remarks can be concluded from the results.

First, reducing the factors that negatively affect the delay-to-distance mapping process (such as TIVs [46]) is sufficient to provide the ability of relying on the mapped distances for location verification. CPV leverages several heuristics to reduce such factors, e.g., iterating the delay-measurement process and using multiple delay-estimation protocols. Its accuracy improves upon applying these heuristics.

Second, relying on OWDs instead of RTTs enhances the accuracy of the location verification process. Since common OWD-estimation protocols [24] fail to consider adversarial environments, a novel protocol called mp is presented. It promises higher estimation accuracy than halving the RTT, yet requires less client cooperation than OWAMP-like protocols.

The design of CPV provides several security and deployability advantages. It overcomes IP-hiding tactics typically carried out using middleboxes, since delays are measured over the client's application layer. Additionally, CPV requires no client-side changes, and no extra software is needed; the client's current browsing experience is retained as the verification process runs in the browser. These advantages and the real-world evaluation results highlight CPV's potential for practical adoption.

Acknowledgements

The second author acknowledges support from the Natural Sciences and Engineering Research Council of Canada (NSERC) through a Discovery Grant. The third author acknowledges funding from NSERC for both his Canada Research Chair in Authentication and Computer Security, and a Discovery Grant.

Appendix

The three claims made earlier in the paper are now proved.

Notation. The notation $\bigcirc XY(k)$ refers to the ellipse determined by the foci X and Y whose major axis is k meters long; $\overline{AB}$ for the *length* of line segment AB; and $\overleftrightarrow{XY}$ refers to the straight line passing by the points X and Y.

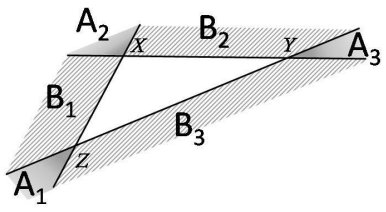


Fig. 14. Regions $A = A_1 \cup A_2 \cup A_3$ and $B = B_1 \cup B_2 \cup B_3$ outside $\triangle XYZ$.

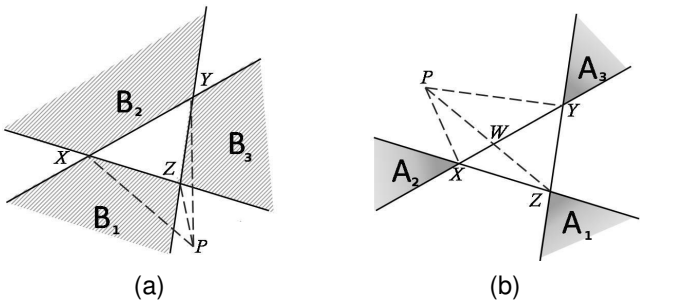


Fig. 15. If P is outside $\triangle XYZ$, the sum of the areas of $\triangle XYP$, $\triangle XPZ$ and $\triangle ZPY$ will be larger than the area of $\triangle XYZ$.

Consider $\triangle XYZ$ in Fig. 14. Regions A_1, A_2 and A_3 are those outside $\triangle XYZ$ delimited by the pairs $(\overleftrightarrow{XZ}, \overleftrightarrow{YZ})$, $(\overleftrightarrow{XY}, \overleftrightarrow{XZ})$ and $(\overleftrightarrow{XY}, \overleftrightarrow{YZ})$ respectively, such that none of $\triangle XYZ$'s exterior angles belong to A_1, A_2 or A_3. Regions B_1, B_2 and B_3 are those outside $\triangle XYZ$ delimited by the region pairs (A_1, A_2), (A_2, A_3) and (A_3, A_1) respectively. A point P outside $\triangle XYZ$ will either fall in region $A = A_1 \cup A_2 \cup A_3$ or $B = B_1 \cup B_2 \cup B_3$.

Proof of Claim 1
Recall Claim 1: Let P be a point in the Cartesian plane, and let $\triangle XYZ$ be the triangle determined by the points X, Y and Z. If P is strictly outside $\triangle XYZ$, then the sum of the areas of $\triangle XYP$, $\triangle XPZ$ and $\triangle PYZ$ is greater than the area of $\triangle XYZ$.

First, assume that P is in region A; then:

Claim 4. If P is in region A, then the area of one of the triangles $\triangle XYP$, $\triangle XPZ$ or $\triangle PYZ$ will be larger than the area of $\triangle XYZ$.

Proving claim 4 suffices to prove claim 1 for region A because if the area of only one triangle by itself exceeds the area $\triangle XYZ$, then the sum of the areas of the three triangles ($\triangle XYP$, $\triangle XPZ$ and $\triangle PYZ$) will definitely exceed the area of $\triangle XYZ$. To prove claim 4, assume that P is in region A_1, as shown in Fig. 15a. In this case, the one triangle (referred to in claim 4) whose area is larger than that of $\triangle XYZ$ is $\triangle XYP$. The proof follows.

Proof:
Since region A_1 is bound by the straight line pair $(\overleftrightarrow{XZ}, \overleftrightarrow{YZ})$.
Therefore $\angle YXP > \angle YXZ$ and $\angle XYP > \angle XYZ$.
Therefore Z is inside $\triangle XYP$.
Since line segment XY is shared between $\triangle XYZ$ and $\triangle XYP$.
Therefore $\triangle XYZ \subset \triangle XYP$.
Therefore $area(\triangle XYZ) < area(\triangle XYP)$. □

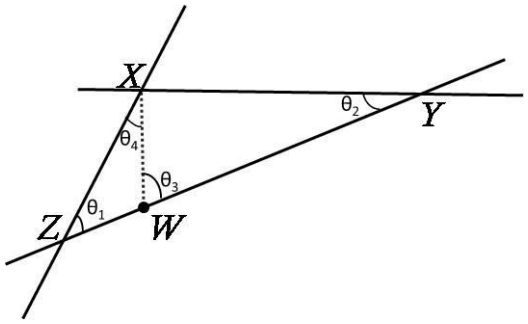


Fig. 16. If $\overline{XZ} \leq \overline{XY}$ and W is inside $\triangle XYZ$, then $\overline{XW} \leq \overline{XY}$.

Note that an analogous proof holds if P is in A_2 or A_3. For region B:

Claim 5. If P is in region B, then the sum of the areas of two of the three triangles $\triangle XYP$, $\triangle XPZ$ or $\triangle PYZ$ will be larger than the area of $\triangle XYZ$.

Again, proving claim 5 suffices to prove claim 1 for region B because the sum of the areas of the three triangles ($\triangle XYP$, $\triangle XPZ$ and $\triangle PYZ$) will definitely exceed the area of $\triangle XYZ$ if the areas of two of the three triangles together exceed the area $\triangle XYZ$. To prove claim 5, assume that P is in region B_2, as shown in Fig. 15b; line segment PZ intersects XY in W. In this case, the two triangles (referred to in claim 5) are $\triangle XPZ$ and $\triangle ZPY$. The proof follows.

Proof:
Since P, W and Z are collinear, W is between P and Z, and
Since line segment XZ is shared between $\triangle XWZ$ and $\triangle XPZ$
Therefore $\triangle XWZ \subset \triangle XPZ$
Similarly, $\triangle ZWY \subset \triangle ZPY$
Therefore $(\triangle XWZ \cup \triangle ZWY) \subset (\triangle XPZ \cup \triangle ZPY)$
Therefore $\triangle XYZ \subset (\triangle XPZ \cup \triangle ZPY)$.
Therefore $area(\triangle XYZ) < area(\triangle XPZ) + area(\triangle ZPY)$. □

Analogous proof holds if P is in B_1 or B_3. This concludes the proof to Claim 1.

Proof of Claim 2
Recall Claim 2: Let W be a point in the Cartesian plane, and let $\triangle XYZ$ be the triangle determined by the points X, Y and Z such that $\overline{XZ} \leq \overline{XY}$. If $\overline{XW} > \overline{XY}$, then W is strictly outside of $\triangle XYZ$.

This Claim can be rewritten as:

Claim 6. Let W be a point in the Cartesian plane, and let $\triangle XYZ$ be the triangle determined by the points X, Y and

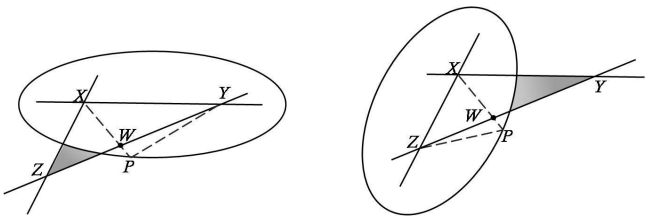


Fig. 17. When $P \in B_3$, then $\triangle XYZ \subset \{\bigcirc XY(\overline{XP}+\overline{PY}) \cup \bigcirc XZ(\overline{XP}+\overline{PZ})\}$.

Z; $\overline{XZ} \leq \overline{XY}$. If W is inside $\triangle XYZ$, then $\overline{XW} \leq \overline{XY}$.

which is the logical transposition $(P \rightarrow Q) \vdash (\neg Q \rightarrow \neg P)$ of Claim 2, where P is the event "$\overline{XW} > \overline{XY}$", and Q is the event "W is strictly outside of $\triangle XYZ$". The following proves that $\overline{XW} \leq \overline{XY}$ holds when $\overline{XW}$ is the maximum that maintains W inside $\triangle XYZ$, which is when W lies on line segment YZ (see Fig. 16).

Proof:
Since $\overline{XZ} \leq \overline{XY}$
Therefore $\theta_2 \leq \theta_1$.
Since W lies on line segment YZ
Therefore $\theta_1 + \theta_4 = \theta_3$.
Therefore $\theta_1 \leq \theta_3$.
Therefore $\theta_2 \leq \theta_3$.
Therefore $\overline{XW} \leq \overline{XY}$. □

Proof of Claim 3
Recall Claim 3: Let P be a point in the Cartesian plane, and let $\triangle XYZ$ be the triangle determined by the points X, Y and Z. If P is strictly outside $\triangle XYZ$, then increasing the sums $\overline{XP}+\overline{PZ}$, $\overline{XP}+\overline{PY}$ or $\overline{YP}+\overline{PZ}$ without reducing at least one of the other sums cannot place P inside $\triangle XYZ$.

Similar to the proof of Claim 1, the proof of Claim 3 is split into two parts: when $P \in A$ and when $P \in B$. For part one, first assume that $P \in A_1$. In this case, according to the isoperimetric inequality, $\overline{XP}+\overline{PY}$ must be greater than $\overline{XZ}+\overline{ZY}$ because they both have the same starting and ending points, X and Y. Therefore, it is impossible to move P inside $\triangle XYZ$ without decreasing $\overline{XP}+\overline{PY}$. Analogous argument applies for regions A_2 and A_3.

Now to the case where $P \in B$. First assume that $P \in B_3$ as shown in Fig 17. If $\triangle XYZ \subset \{\bigcirc XY(\overline{XP}+\overline{PY}) \cup \bigcirc XZ(\overline{XP}+\overline{PZ})\}$,[8] is proved, then P cannot move to inside $\triangle XYZ$ without reducing $\overline{XP}+\overline{PY}$ or $\overline{XP}+\overline{PZ}$ because the sum of the lengths from any point on the ellipse to its pair of foci is constant; hence, the sum of the lengths from any point inside the ellipse to its pair of foci is less than that to any point on the ellipse.

Assume that $\triangle XYZ$ is split into two: $\triangle XYW$ and $\triangle XWZ$, where W is the intersection of line segments XP and YZ. Then, proving that $\triangle XYW \subset \bigcirc XY(\overline{XP}+\overline{PY})$ is as follows (see Fig. 17a).

8. Note that $\triangle \subset \bigcirc$ if $\forall p \in \triangle$, $p \in \bigcirc$.

Proof:
Since X is a focus of the ellipse; P is a point on the ellipse; X, W and P are collinear; and $P \notin \triangle XYZ$
Therefore W is inside the ellipse.
Since Y is a focus of the ellipse
Therefore line segments XW, WY and XY are inside the ellipse.
Therefore $\triangle XYW \subset \bigcirc XY(\overline{XP}+\overline{PY})$. □

Analogous proof applies to $\triangle XWZ \subset \bigcirc XZ(\overline{XP}+\overline{PZ})$ (Fig. 17b). Therefore, when $P \in B_3$, it is impossible to move P inside $\triangle XYZ$ without reducing the summation $\overline{XP}+\overline{PY}$ or $\overline{XP}+\overline{PZ}$. The remaining regions of B can be proved in the same manner. Therefore, whenever $P \in B$, then $\triangle XYZ \subset \{\bigcirc XY(\overline{XP}+\overline{PY}) \cup \bigcirc YZ(\overline{YP}+\overline{PZ}) \cup \bigcirc XZ(\overline{XP}+\overline{PZ})\}$. This concludes the proof.

REFERENCES

[1] A. M. Abdou, A. Matrawy, and P. C. van Oorschot, "Location Verification on the Internet: Towards Enforcing Location-aware Access Policies Over Internet Clients," in *IEEE CNS*, Oct. 2014.
[2] J. Burnett, "Geographically Restricted Streaming Content and Evasion of Geolocation: the Applicability of the Copyright Anticircumvention Rules," *MTTLR*, vol. 19, no. 2, 2013.
[3] I. Polakis, S. Volanis, E. Athanasopoulos, and E. P. Markatos, "The Man Who Was There: Validating Check-ins in Location-based Services," in *ACM ACSAC*, 2013.
[4] M. Trimble, "The Future of Cybertravel: Legal Implications of the Evasion of Geolocation," *Fordham IPLJ*, vol. 22, 2011.
[5] J. A. Muir and P. C. van Oorschot, "Internet geolocation: Evasion and counterevasion," *ACM Comput. Surv.*, vol. 42, no. 1, pp. 4:1–4:23, 2009.
[6] A. Popescu, "Geolocation API Specification," October 2013. [Online]. Available: http://www.w3.org/TR/geolocation-API/
[7] D. Hu and C.-L. Wang, "GPS-Based Location Extraction and Presence Management for Mobile Instant Messenger," in *LNCS EUC*, 2007.
[8] P. A. Zandbergen, "Accuracy of iPhone Locations: A Comparison of Assisted GPS, WiFi and Cellular Positioning," *Transactions in GIS*, vol. 13, pp. 5–25, 2009.
[9] "MaxMind - IP Geolocation and Online Fraud Prevention." [Online]. Available: https://www.maxmind.com
[10] M. Casado and M. Freedman, "Peering Through the Shroud: The Effect of Edge Opacity on IP-based Client Identification," in *USENIX NSDI*, 2007.
[11] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," in *USENIX Security*, 2004.
[12] S. Laki, P. Mátray, P. Hága, T. Sebók, I. Csabai, and G. Vattay, "Spotter: A model based active geolocation service," in *IEEE INFOCOM*, 2011.
[13] M. J. Arif, S. Karunasekera, and S. Kulkarni, "GeoWeight: Internet Host Geolocation Based on a Probability Model for Latency Measurements," in *ACSC*. Australian Computer Society, 2010.
[14] P. Gill, Y. Ganjali, B. Wong, and D. Lie, "Dude, where's that IP? Circumventing measurement-based IP geolocation," in *USENIX Security*, 2010.
[15] N. Sastry, U. Shankar, and D. Wagner, "Secure Verification of Location Claims," in *ACM WiSec*, 2003.
[16] S. Capkun and J.-P. Hubaux, "Secure positioning of wireless devices with application to sensor networks," in *IEEE INFOCOM*, 2005.
[17] Z. Dong, R. D. Perera, R. Chandramouli, and K. Subbalakshmi, "Network measurement based modeling and optimization for IP geolocation," *Computer Networks*, vol. 56, no. 1, pp. 85–98, 2012.
[18] A. Ziviani, S. Fdida, J. F. de Rezende, and O. C. M. Duarte, "Improving the accuracy of measurement-based geographic location of Internet hosts," *Computer Networks*, vol. 47, no. 4, pp. 503–523, 2005.
[19] R. Landa, R. G. Clegg, J. T. Araujo, E. Mykoniati, D. Griffin, and M. Rio, "Measuring the Relationships between Internet Geography and RTT," in *IEEE ICCCN*, 2013.
[20] A. Pathak, H. Pucha, Y. Zhang, Y. C. Hu, and Z. M. Mao, "A measurement study of Internet delay asymmetry," in *Springer PAM*, 2008.
[21] A. M. Abdou, A. Matrawy, and P. C. van Oorschot, "Accurate One-Way Delay Estimation with Reduced Client-Trustworthiness," *IEEE Commun. Lett.*, vol. 19, no. 5, pp. 735–738, 2015.

[22] I. Cunha, R. Teixeira, D. Veitch, and C. Diot, "DTRACK: A System to Predict and Track Internet Path Changes," *IEEE/ACM Trans. Netw.*, vol. 22, no. 4, pp. 1025–1038, 2014.

[23] B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman, "PlanetLab: An Overlay Testbed for Broad-coverage Services," *SIGCOMM Comput. Commun. Rev.*, vol. 33, no. 3, pp. 3–12, 2003.

[24] S. Shalunov, B. Teitelbaum, A. Karp, J. Boote, and M. Zekauskas, "A One-way Active Measurement Protocol (OWAMP)," RFC 4656 (Proposed Standard), Sep. 2006.

[25] G. Wang, B. Zhang, and T. Ng, "Towards network triangle inequality violation aware distributed systems," in *ACM IMC*, 2007.

[26] B. Wong, I. Stoyanov, and E. G. Sirer, "Octant: a comprehensive framework for the geolocalization of Internet hosts," in *USENIX NSDI*, 2007.

[27] B. Gueye, A. Ziviani, M. Crovella, and S. Fdida, "Constraint-based geolocation of Internet hosts," *IEEE/ACM Trans. Netw.*, vol. 14, no. 6, pp. 1219–1232, 2006.

[28] E. Katz-Bassett, J. P. John, A. Krishnamurthy, D. Wetherall, T. Anderson, and Y. Chawathe, "Towards IP geolocation using delay and topology measurements," in *ACM IMC*, 2006.

[29] F. Dabek, R. Cox, F. Kaashoek, and R. Morris, "Vivaldi: A decentralized network coordinate system," in *ACM SIGCOMM*, 2004.

[30] B. Donnet, B. Gueye, and M. A. Kaafar, "A survey on network coordinates systems, design, and security," *IEEE Commun. Surv. & Tut.*, vol. 12, no. 4, pp. 488–503, 2010.

[31] F. Girlich, M. Rossberg, G. Schaefer, T. Boehme, and J. Schreyer, "Bounds for the Security of the Vivaldi Network Coordinate System," in *IEEE NetSys*, 2013.

[32] S.-H. Yook, H. Jeong, and A.-L. Barabási, "Modeling the Internet's large-scale topology," *NAS of the USA*, vol. 99, no. 21, pp. 13 382–13 386, 2002.

[33] L. Subramanian, V. N. Padmanabhan, and R. H. Katz, "Geographic properties of Internet routing," in *USENIX ATC*, 2002.

[34] A. Zeitoun, C.-N. Chuah, S. Bhattacharyya, and C. Diot, "An AS-level study of internet path delay characteristics," in *IEEE GLOBECOM*, 2004.

[35] I. Fette and A. Melnikov, "The WebSocket Protocol," RFC 6455 (Proposed Standard), Dec. 2011.

[36] W. Li, R. K. Mok, R. K. Chang, and W. W. Fok, "Appraising the Delay Accuracy in Browser-based Network Measurement," in *ACM IMC*, 2013.

[37] Y. Zhang and N. Duffield, "On the Constancy of Internet Path Properties," in *ACM IMW*, 2001.

[38] C. Lumezanu, R. Baden, N. Spring, and B. Bhattacharjee, "Triangle inequality and routing policy violations in the internet," in *Springer PAM*, 2009.

[39] P. Hsu and H. Robbins, "Complete convergence and the law of large numbers," *NAS of the USA*, vol. 33, no. 2, pp. 25 – 31, 1947.

[40] M. Crovella and B. Krishnamurthy, *Internet Measurement: Infrastructure, Traffic and Applications*. John Wiley & Sons, 2006.

[41] L. De Vito, S. Rapuano, and L. Tomaciello, "One-Way Delay Measurement: State of the Art," *IEEE Trans. Instrum. Meas.*, vol. 57, no. 12, pp. 2742–2750, Dec 2008.

[42] J. Liu, "A novel method for estimating the variable and constant components of one-way delays without using the synchronized clocks," in *ICNC*, Feb 2014, pp. 1028–1033.

[43] O. Gurewitz, I. Cidon, and M. Sidi, "One-way delay estimation using network-wide measurements," *IEEE/ACM Trans. Netw.*, vol. 14, no. SI, pp. 2710–2724, 2006.

[44] A. Vakili and J. Gregoire, "Accurate One-Way Delay Estimation: Limitations and Improvements," *IEEE Trans. Instrum. Meas.*, vol. 61, no. 9, pp. 2428–2435, 2012.

[45] D. Mills, J. Martin, J. Burbank, and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification," RFC 5905 (Proposed Standard), Oct. 2010.

[46] Y. Zhang and H. Zhang, "Triangulation Inequality Violation in Internet Delay Space," in *Advances in Computer Science and Information Engineering*. Springer, 2012, vol. 169, pp. 331–337.

[47] A. M. Abdou, A. Matrawy, and P. C. van Oorschot, "Taxing the Queue: Hindering Middleboxes from Unauthorized Large-Scale Traffic Relaying," *IEEE Commun. Lett.*, vol. 19, no. 1, pp. 42–45, 2015.

[48] S. Brands and D. Chaum, "Distance-Bounding Protocols," in *Advances in Cryptology*. Springer, 1994, vol. 765, pp. 344–359.

[49] Z. Zhu and G. Cao, "APPLAUS: A Privacy-Preserving Location Proof Updating System for Location-based Services," in *IEEE INFOCOM*, 2011.

[50] W. Luo and U. Hengartner, "VeriPlace: A Privacy-Aware Location Proof Architecture," in *ACM SIGSPATIAL*, 2010.

[51] F. Olumofin, P. Tysowski, I. Goldberg, and U. Hengartner, "Achieving Efficient Query Privacy for Location Based Services," in *Springer PET*, 2010.

AbdelRahman Abdou is a PhD student in the Department of Systems and Computer Engineering at Carleton University. His research interests include location-aware security, Future Internet Architectures (FIAs), and using Internet measurements to solve problems related to Internet security.

Ashraf Matrawy is an Associate Professor of the School of Information Technology at Carleton University. He is a senior member of the IEEE and serves on the editorial board of the IEEE Communications Surveys and Tutorials journal. He has served as a technical program committee member of IEEE CNS, IEEE ICC, IEEE Globecom, IEEE LCN, and IEEE/ACM CC-GRID. He is also a Network co-Investigator of Smart Cybersecurity Network (SERENE-RISC). His research interests include reliable and secure computer networking, software defined networking and cloud computing.

Paul C. van Oorschot is a Professor of Computer Science at Carleton University, and the Canada Research Chair in Authentication and Computer Security. He was the program chair of USENIX Security 2008, NDSS 2001-2002, a co-author of the Handbook of Applied Cryptography and a past editor of IEEE TDSC, IEEE TIFS, and ACM TISSEC. His research interests include authentication and Internet security.