Simple Location-Based One-time Passwords

bringing location to the cloud in a secure and simple way

Roland van Rijswijk-Deij
Radboud University Nijmegen, The Netherlands
and SURFnet bv, Utrecht, The Netherlands
roland.vanrijswijk@surfnet.nl

ABSTRACT

Implicitly, the location of a user has played a role in authentication for a long time. For example, many organizations use IP-based access control rules to restrict access to certain services. The move of business processes to cloud services combined with the increasing mobility of users makes this paradigm problematic. Researchers try to bridge this gap by explicitly making location a key factor in multi-factor authentication systems. Most solutions proposed to date are complex and/or require significant changes to standard infrastructures. This paper introduces a novel approach to using location in authentication that is simple to implement, relies on existing open standards, works with off-the-shelf components and can be implemented at low cost. The proposed solution leverages features of the new Bluetooth Low Energy specification and is compatible with Apple's recently introduced iBeacon technology. The basic premise of the proposed system is to replace the static identifiers in the iBeacon technology, that report a location, by location-specific one-time passwords that can be used as an additional factor in an authentication system.

Categories and Subject Descriptors

D.4.6 [**Operating Systems**]: Security and Protection—*multi-factor authentication*; K.6.5 [**Management of Computing and Information Systems**]: Security and Protection—*multi-factor authentication*

Keywords

multi-factor authentication, iBeacon, Bluetooth Low Energy, context-based authentication, location-based security

1. INTRODUCTION

The past five years have seen a revolution in the way we use the Internet on-the-go; the rapid proliferation of smartphones and - even more recently - tablet computers has revolutionized the way we interact online. The security industry has embraced this storm of new devices that users now have by increasingly leveraging *something the user* ***already*** *has* as one of the factors in multi-factor authentication systems. Most main-stream vendors of two-factor authentication systems now have app versions of their traditional security tokens (e.g. Vasco DigiPass for Mobile[1] or SafeNet MobilePASS[2]).

Devices like smartphones offer other interesting new possibilities to give accurate information about the location of the device. Uptake of these capabilities for security purposes has been much more lackluster. Although there has been quite a bit of research into using location as a factor in authentication, the practicality of the approaches taken so far can be called to question. This is partly because most solutions rely on complex calculations of the region in which a user resides, calculations which in turn depend on accurate positional information about the user. This positional information is often based on GPS or signal triangulation of cell phone towers outdoors, and on Wi-Fi access point signal strength indoors. Radio signal-based position determination requires a priori knowledge of the position of aerials which implies a significant investment in time and planning if these systems are to be of use in real life.

Smartphone vendors have come to realize that indoor location information is currently inaccurate, which hampers its use in applications. For example, Apple has introduced what it calls "iBeacon" into the most recent release (version 7) of its iOS mobile operating system. The iBeacon technology relies on Bluetooth Low Energy to broadcast beacon signals that can be used by applications on Apple's portable devices to perform accurate indoor positioning. Apple markets this technology as a way for merchants to send location-based advertising to users or as a way for museum exhibitors to enable location-based interaction between visitors and certain museum displays. As with many newly introduced technologies, the potential for innovation of a system like iBeacon exceeds the original intentions of the manufacturer. Within months of releasing this technology, independent vendors have already started using this technology to leverage the location of a user as a ground for authorization decisions (e.g. ExactEditions magazine promotion technology[3]).

[1] `http://www.vasco.com/products/client_products/software_digipass/digipass_for_mobile.aspx`

[2] `http://www.safenet-inc.com/products/data-protection/two-factor-authentication/mobilepass/`

[3] `http://blog.exacteditions.com/2013/11/29/`

Contribution. The contribution of this paper is that it takes the iBeacon technology and uses it to enhance security by adding location as a factor to existing, open standards-based two-factor authentication systems (such as, for instance, systems based on the OCRA standard [1]). The proposed system will not only work with smartphones but also tablets and other devices that support Bluetooth Low Energy. The novelty of the proposed system not only lies in applying the iBeacon technology for security purposes, but also in not using the exact position of a user as a location factor but rather relying on proximity to a beacon as a factor.

2. RELATED WORK

In a broader sense, using location as a factor in authentication is a particular form of context-aware authentication. The prevalent definitions of "context" and "context-aware" used when discussing context-aware systems were introduced by Dey [2]. Although Dey's work is geared towards applications that use context to tailor their behavior such that the user experience improves, his definitions are also applicable to using context for authentication. A good survey of the use of context in authentication is given by Bertino and Kirkpatrick [3]. They identify two ways in which location can play a role in authentication: either as a factor in multi-factor authentication (which applies to the system proposed in this paper) or as a determinant in deciding what security policy to apply in a certain context (IP-based authentication referred to in the abstract of this paper is an example of this).

A comprehensive overview of two-factor authentication approaches is given by Van Rijswijk and Van Dijk in [4]; apart from providing an overview of traditional token-based solutions they also discuss mobile device-based solutions and introduce an app-based approach that leverages the features of modern smartphones. The system they propose, called "tiqr", is open source, and as such a good candidate to be extended to work with the approach proposed in this paper.

Zhang et al. [5] describe a system that uses the geolocation features of smartphone platforms to create a location-based authentication scheme. The system they propose relies primarily on GPS for location information, but they also augment location information by gathering information about nearby Wi-Fi access points. The scheme they describe does not rely on specialized infrastructure and the authors claim that it can easily be integrated into existing authentication schemes by means of plug-ins; they do not, however, elaborate on what these plug-ins could be or provide a proof-of-concept implementation. They also state that their solution takes steps to protect the privacy of the user's location but do not elaborate on how this goal is achieved.

A markedly different approach to using location in authentication is taken by Hsieh and Leu [6]. They propose a rather complex scheme where various measurements of the user's location over time (based on GPS data) are used to predict where the user is likely to be when they wish to access a service. That information is combined with the time of login to generate a time and location-specific one-time password (OTP).

promotion-by-ibeacons/

More closely related to the system introduced in this paper is Jansen and Korolev's [7] effort. They describe how Bluetooth beacons can be used to set up a perimeter within which users are granted additional privileges based on a policy that is also conveyed by means of the beacons used to establish the perimeter. Their work differs from the work in this paper in two ways. First, in their approach authentication is local to the device, i.e. access control decisions are taken locally on the device (e.g. allowing or disallowing users to run certain applications based on the device's location). They do not use the beacon as an authentication factor to a remote server through the mobile device. Secondly, rather than using the faster and more energy-efficient Bluetooth Low Energy technology used in this paper, their approach relies on the classic Bluetooth technology; this makes their approach less suitable for energy-constrained devices like smartphones. Another notable disadvantage of using classic Bluetooth is that it requires pairing of mobile devices and beacons, which adversely impacts the user experience.

The concept of constrained channels, introduced by Kindberg et al. [8] can be applied - in abstracto - to the system proposed in this paper. In their work they model communication channels that can only be used under certain constraints. Users wanting to send information over a communication channel must meet the constraints before they can use the channel. The most important example used by Kindberg et al. is location; they sketch a scenario where a user must be in proximity of range-bounded channel (they give the example of a Bluetooth transceiver) and use that information to establish the location of a user for authentication purposes.

Finally, Cho et al. [9] describe a system that leverages the use of access point-specific keys. They describe how the combination of several such keys, taken from Wi-Fi access points (APs) a user's device sees, can be used to determine the location of the user by deriving a location specific key from the AP keys. This derived key can then be used to moderate network access for the user, only allowing those users that have derived a valid key access to the network. The example they provide in their paper illustrates how the system can be used to only grant access to users of the wireless network in a cybercafé if they are actually inside, implying that they mostly see their work as a way to regulate network access. It is, however, easy to see how their work could be extended for use as a means of using location as a factor in authentication. Where their approach compares less favourably with the approach taken in this paper is that it requires extensive interaction (that may be restricted to privileged administrator accounts on some devices) with the wireless network stack on the mobile device as well as the ability to manipulate the behaviour of wireless access points.

3. SYSTEM ARCHITECTURE

This section introduces a new proposal for using location as an additional factor in authentication. The majority of traditional multi-factor authentication systems rely on two factors: *something the user knows* and *something the user has*. When a user authenticates to a service using such a system, this translates into authenticating a *specific user* with a *specific device (token)*.

The solution proposed in this paper adds an additional factor to the set, namely *something the user is close to*. By tying this factor into an existing system based on the something the user knows+has paradigm, the authentication now is of a *specific user* with a *specific device* near a *specific location*.

What further differentiates the solution described here from other approaches described in Section 2 is that it:

- is simple to implement;
- relies on existing open standards;
- works with unmodified off-the-shelf mobile devices;
- can be implemented with cheap hardware for the location beacons.

3.1 Introducing iBeacons and Bluetooth Low Energy

Apple's iBeacon technology leverages features of the Bluetooth Low Energy (BLE) technology introduced with the fourth major version of the Bluetooth protocol specification. The system proposed in this paper is inspired by Apple's iBeacon technology in the sense that it is compatible with the iBeacon implementation and can thus easily be integrated in an iOS application. The basic concept, however, can be implemented on top of any BLE stack with a bit more effort on the side of the application developer[4].

BLE is a completely new protocol that is not backward compatible with the traditional Bluetooth protocol stack. BLE operates in the 2.4GHz range of the frequency spectrum. It defines a wide range of new application profiles that devices can implement, a discussion of which is beyond the scope of this paper. According to the Bluetooth group, BLE-enabled devices can operate over a distance of 50m, which can be influenced further by varying the transmission power of the device to bring the maximum distance down.

Devices that implement BLE advertise[5] their presence at regular intervals[6] on three bands that are spread across the spectrum to prevent interference with other radio signals (e.g. Wi-Fi). As part of these advertisement broadcasts, devices can transmit up to 31 bytes of custom payload data[7]. In Apple's iBeacon system, this customizable section of the advertisement packet is used to transmit three data elements[8]:

- a 128-bit Universally Unique IDentifier (UUID);
- a 16-bit "major" location value;
- a 16-bit "minor" location value.

The intended use of these values in the iBeacon system is as follows: the UUID is used to identify one or more beacons belonging to a group; the iOS `CoreLocation` framework allows applications to scan for beacons that broadcast a specific UUID. The *major* and *minor* value can be used to further identify a specific beacon that is part of the group identified by the UUID. A practical example of the use would be to augment an app for a museum exhibition with location information. The museum would then use a single UUID for the exhibition that the app knows about and assign major/minor values to rooms and specific exhibits, such that proximity to a beacon would allow the app to provide the user with more information about the exhibit they are looking at.

How applications that use iBeacons choose between two or more beacons that are nearby is left to the application implementer; in addition to looking for beacons with specific UUIDs, the `CoreLocation` framework also provides rough proximity information about beacons, based on the radio signal strength indicator (RSSI). This could be used as a way to choose the closest beacon.

3.2 System overview

The examples given in the previous section show iBeacons in use in a static manner, i.e. the UUID, major and minor value are fixed for a specific beacon. The combination of values is used to relate information about a location to applications on a mobile device. The system proposed in this paper works differently; rather than using fixed values for major/minor these two values will be used to convey dynamic data to a mobile device that can be used in an authentication context. More specifically, the major and minor values will be used to transmit time-based one-time passwords from a beacon to a mobile device.

Figure 1 shows a high-level overview of the proposed system. The components shown are:

- **OTP beacon** – this is a BLE-enabled device that transmits time-based one-time passwords at regular intervals as part of its normal device advertisements and compatible with Apple's iBeacon technology. The OTPs are based on a beacon/location-specific key k_b (more on the protocol in Section 3.3 below).
- **Smartphone**[9] – this is a regular smartphone that runs an OTP app that can generate OTPs based on a user-specific key k_p that is tied to the phone.
- **Server** – the server authenticates the user; it knows both k_b and k_p which it needs to verify any OTPs submitted by a user wishing to authenticate.

[4] e.g. Radius Networks' iBeacon library for Android `http://developer.radiusnetworks.com/ibeacon/android/`

[5] The term "advertise" in the context of BLE means making the device known to others for the purpose of device discovery by broadcasting a signal and is not to be confused with the use of iBeacons for the purpose of commercial advertising.

[6] Suggestions on how frequently a device should transmit these advertisements are provided in the Bluetooth 4.0 specifications.

[7] Source: `http://www.eetimes.com/document.asp?doc_id=1278927`

[8] Source: `http://www.theregister.co.uk/2013/11/29/feature_diy_apple_ibeacons/`

[9] Wherever the term smartphone is used in the paper, this can also be read as "any mobile device with BLE" (e.g. tablet, laptop)

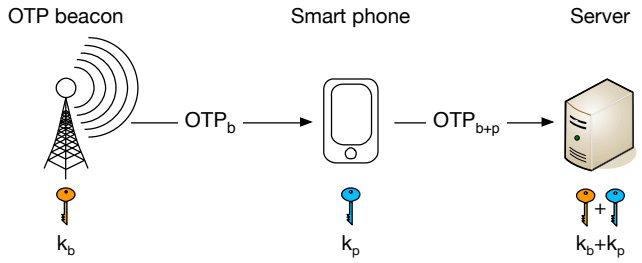

Figure 1: System component overview

Abstractly, an authentication flow looks like this:

1. A user wishes to authenticate to the Server.
2. The user activates the OTP authentication app on their Smartphone.
3. The app scans for a nearby beacon and receives an OTP_b from the beacon when it is detected through its regular advertisement broadcasts.
4. The app now generates an OTP_{b+p} based on its device- and user-specific secret; this OTP_{b+p} also takes OTP_b received from the beacon as input.
5. The app transmits its OTP_{b+p} together with the beacon's OTP_b to the Server.
6. The server verifies both OTPs and grants the user access if they verify successfully.

The end result is that a *specific user* with a *specific device* at a *specific location* is authenticated, since authentication is only possible if the user is near a beacon that is transmitting OTPs and if the user is in possession of a device with their user-specific key and account on it.

The next section gives more details on the exact protocol.

3.3 Protocol

Note that throughout the protocol description subscript letters are used to indicate that source of values: a subscript b signifies that the value belongs to or originated at the beacon, a subscript p indicates the smartphone and the combination $b + p$ signifies that it is a combined value derived from data from both the beacon as well as the smartphone.

3.3.1 Beacon OTP protocol

Beacons will generate time-based one-time passwords using the TOTP [10] algorithm. They will use a beacon-specific key k_b in the generation of the OTP and will also need a reliable clock[10] that is synchronised (within certain tolerances) with the clock in the authentication server. The beacon will generate a new OTP every n seconds, where $n = 30$ is the default (recommended in [10], Section 5.2, more on this in Section 4.2).

[10]Having a reliable clock is a realistic requirement; most commercially available OTP tokens use time-based OTP and are simple, battery-operated devices.

Each beacon will also have an optional 8-bit beacon-specific identifier i_b that can be used to identify a specific beacon within a group (remember from Section 3.1 that a group of beacons is identified using a UUID).

To transmit the OTP, the beacon now does the following:

1. Generate an OTP according to the TOTP algorithm as specified in [10], using the beacon-specific key k_b and truncated to a 6-digit integer value following the steps set out in [11]; this is the current OTP.
2. Encode the 6 digits of the current OTP as binary-coded decimals (BCD), giving a 24-bit value OTP_{BCD}.
3. Create the concatenation $OTP_b = i_b \| OTP_{BCD}$, a 32-bit value.
4. Split OTP_b into two values $OTP_{b(h)}$, the most significant 16 bits of OTP_b and $OTP_{b(l)}$, the least significant 16 bits of OTP_b.
5. Set the *major* advertising value of the beacon to be $OTP_{b(h)}$ and the *minor* advertising value of the beacon to be $OTP_{b(l)}$.

To illustrate this with an example, assume a beacon with i_b = `0xAB` and the current OTP value equal to 672310. In this case, OTP_{BCD} = `0x00672310`. Consequently, OTP_b = `0xAB672310` meaning that the beacon will transmit as *major* value $OTP_{b(h)}$ = `0xAB67` and as *minor* value $OTP_{b(l)}$ = `0x2310`.

3.3.2 Smartphone OTP protocol

As an example, we consider adding location information as an extra factor to a smartphone OTP application that uses the OCRA protocol [1]. OCRA is a challenge-response based OTP authentication system. OCRA OTPs are generated based on a device-specific key tied to the device and the user's account, k_p; access to this key is assumed to be protected using a pass phrase or a PIN. One of the inputs to the OCRA algorithm when generating an OTP from a server-specified challenge is the value S, which is session specific (e.g. information about the TLS session of a web application as suggested in `IC8` in Section 8.2 of [1]).

The location-specific OTP_b will be appended to the session data input S to the OCRA protocol in order to tie the OTP_b value the smartphone receives from the beacon to the challenge/response OTP_{b+p} generated by the smartphone.

To illustrate the impact this has on an existing OCRA challenge/response-based smartphone app, here is a list of the changes that would be required:

1. The app will need to scan for beacons transmitting OTPs.
2. It will then have to recreate OTP_b from the *major* and *minor* values transmitted by the beacon by computing $OTP_b = OTP_{b(h)} \| OTP_{b(l)}$.

3. Apart from inputting the challenge C received from the server as part of the OCRA algorithm, it will have to append OTP_b to the regular session data S that is input to the OCRA algorithm.

4. This finally yields OTP_{b+p}, an OTP that ties the users location and device together for authentication.

Note that implementing the system according to this specification means that the system is compatible with the existing OCRA standard. Moreover, it is completely invisible to the user; that is, unless, of course, the access is rejected on the basis of the (wrong) location.

3.3.3 Data sent to the server

Once the steps in the previous two sections have been performed, the following is now available in the smartphone:

> A one-time password $OTP_b \| OTP_{b+p}$, based on *somewhere the user is* (OTP_b which the user can only have by being near a beacon), *something the user has* (OTP_{b+p} which can only be generated using the device-specific key on the user's smartphone) and *something the user knows* (the pass phrase or PIN used to protect the device-specific key on the smartphone).

This location-based one-time password $OTP_b \| OTP_{b+p}$ is transmitted to the server.

3.3.4 Verification by the server

To verify the validity of the data received from the user through their smartphone, the server will perform the following steps:

1. Verify OTP_b using the current time as input; the server will have to accommodate for network latency and a slight time drift between the beacon and the server (see Section 5.2 of [10]).

2. Verify OTP_{b+p} based on the challenge C and session data S sent to the user's smartphone as input for the OCRA algorithm and on the value of OTP_b.

Note that *it is important that both steps above are always performed* regardless of the outcome of step 1; not doing this would leave the system vulnerable to timing attacks.

Figure 2 is a sequence diagram that shows the complete flow of a user logging in using the system proposed in this paper. It shows that the OTP beacon is constantly transmitting new OTPs, and that the smartphone app waits until it detects a beacon OTP before interacting with the server the user is trying to log in to to perform a regular challenge/response authentication.

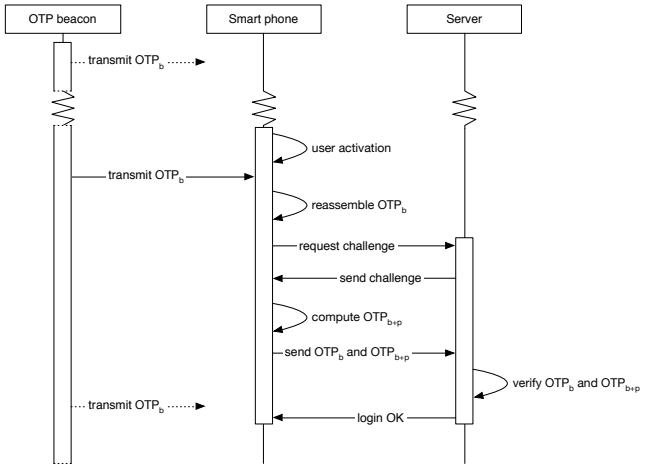

Figure 2: Sequence diagram showing one complete authentication flow

4. DISCUSSION

4.1 Usability

The iBeacon technology that inspired the system proposed in this paper is easy to use, both for developers as well as end users as shown by the experiments described by Smith[8]. Therefore, if the system proposed in this paper is implemented on top of a user-friendly smartphone-based OTP system it will provide a user-friendly solution.

One of the problems that may need to be tackled in terms of usability is dealing with a user being in proximity to multiple beacons. Which beacon's OTP will be used in that case? As was already mentioned in Section 3.1, this choice is left to the application developer in the case of iBeacons. In the system proposed in this paper, multiple solutions can be chosen, for instance limiting the transmission power of the beacon to limit its range or using signal strength indicators (RSSI) to pick the "closest" beacon. It probably depends on the use case which approach is best here. Note that this is a problem only if each beacon has a distinct key and thus transmits different OTPs; if all beacons in an area share the same key this problem will not occur.

4.2 Security analysis

It is trivial to see that the OTPs broadcast by a beacon are not a secure authentication mechanism on their own: there is no protection on the channel on which the beacons broadcast the advertisement data containing the OTP which means that the OTPs can be captured by any device in the beacon's proximity. The system proposed in this paper therefore does not rely on the beacon OTPs as a single factor, but rather combines the OTPs with another factor, the user having a separate device with a device-bound secret. Conveniently, this same device also serves as a receiver for the OTPs broadcast by the beacon.

There are still, however, a number of vulnerabilities inherent in the system. These are:

- **Wormhole attacks** – a wormhole attack (a term also used by Cho et al. [9]) is an attack in which a malicious

entity captures radio traffic from the beacon, pipes it to another location and re-broadcasts the signal there. In the system proposed in this paper, wormhole attacks do not pose a direct threat to the security of the system, since location is not the only factor used for authentication, and in fact, the location-based OTP is strictly tied to an existing shared secret-based OTP on the user's device. An attacker who only performs a wormhole attack does not control this second factor and can thus not impersonate a legitimate user. The only real danger inherent in this kind of attack when applied to the system described in this paper, is that users could be lured to log in at a location where the attacker controls some other aspect of the system which the user utilizes to interact with a service (for instance have malware installed on a kiosk PC that is used to consume a service). There is no easy way to prevent such a wormhole attack, but the likelihood of such an attack is deemed to be quite low; it requires a highly motivated attacker willing to install devices to capture and pipe beacon signals to another location (which incurs costs) and also requires knowledge about the location users are likely to be when attempting to log in to a service. A situation where such an attack would pose a real danger probably warrants the use of other means of protection of both the user as well as the services they access.

- **Denial-of-service through beacon impersonation** – an attacker can perform a denial-of-service attack by introducing an extra beacon controlled by the attacker into the environment that impersonates legitimate beacons (by advertising the UUID of a legitimate beacon) but transmits bogus OTPs (because the attacker does not know the beacon's secret key k_b). This could prevent legitimate users from accessing services and possibly even block their accounts because they will be using invalid OTPs to authenticate.

One thing we have not considered yet is what happens if multiple users try to authenticate from the same location within the time window of a single time-based OTP from the beacon. According to Section 5.2 of [10] TOTPs should not be re-used (to prevent replay attacks within a time window). There are a number of ways of dealing with this:

- The requirement of not re-using TOTPs within a time window can be enforced strictly; the downside of this is that it decreases the usability of the system since users will not be able to authenticate concurrently within the time window of a TOTP. To improve this, the frequency with which new TOTPs are generated can be increased; since users do not have to read the TOTP from a display and type it into a system this is a viable solution.

- The requirement can be relaxed somewhat; instead of outright rejection of a re-used TOTP, the constraint could be revised to only reject re-use of a TOTP by the same user. The risk this introduces is that two users can collude to share TOTPs through some channel where one of the two users can use this TOTP in an authentication while not being in proximity to the beacon.

- Finally, the requirement can be disregarded completely; in addition to the risk of colluding users mentioned before, this also increases the risk of users leaving the beacon area within the time window of the TOTP; we believe, however, that that is not a big problem, since there will have to be a policy to deal with users leaving the beacon area after authenticating in all cases and that policy will cover this case.

In the demonstrator we are building we will implement the last solution; if time permits, we also plan to experiment with increasing the frequency at which TOTPs are generated to see what frequencies can be achieved.

Finally, the system proposed in this paper is based on changes to an existing OCRA-based challenge / response system. We must therefore consider on the one hand if the approach proposed in this paper does not adversely impact the security of the existing OCRA system and on the other hand, it should securely tie location into that existing system. The protocol described in Section 3.3.2 achieves both these goals. The OCRA specification [1] states that the session information S can be used to bind the authentication to specific context information (e.g. information about the application session that a user is authenticating to, or channel binding by including information on the TLS channel over which communication takes place). The proposed protocol leverages this feature of the OCRA specification to bind location to the authentication by appending a beacon-supplied OTP_b to the session information S. To enable the verifier to efficiently validate the final OTP_{b+p} value, OTP_b is also sent directly to the verifier. This does not pose a security risk since an attacker cannot use this value on its own to create a valid OTP_{b+p} (assuming that the attacker does not have access to the device-specific key k_p on the smartphone). Finally, even if an attacker is able to capture a large number of OTP_b values, either by intercepting authentication attempts that include OTP_b or by listening to a beacon for an extended period of time, they learn nothing about the beacon's secret key k_b. The security considerations for TOTP in Section 5.1 of [10] reference the detailed security analysis for HOTP - on which TOTP is based - in Appendix A of [11]; the adversary in this analysis is specifically given unlimited access to valid OTPs generated by the algorithm. The conclusion of the analysis is that an adversary learns nothing of value from valid OTPs and that the best way to attack the system is by brute force.

4.3 Implementation complexity

The implementation complexity (both in terms of software development as well as in the rollout of the system) is one of the biggest drawbacks of existing efforts to incorporate location as a factor in authentication systems (some of which were discussed in Section 2). Most efforts require either a reliable GPS signal (not available indoors) or extensive knowledge of the location of existing radio aerials for things like Wi-Fi access points. Only the system proposed by Jansen and Korolev [7] takes the same approach as proposed in this paper, which is to treat proximity to a transmitting beacon as an indication of location. Where their system dif-

fers, however, is that they rely on "classic" Bluetooth (with higher energy consumption and mandatory device pairing) and that they use proximity beacons to authenticate locally to the device, only allowing users to run certain applications if they are near a beacon.

What really sets the solution proposed in this paper apart is the simplicity with which it can be implemented. The only requirements to make it work are:

- A reliable clock for the time-based OTP implementation on the beacon; this is not a major hurdle as TOTP beacons with the same requirement are already ubiquitously available on the market.
- A power source; also not a major hurdle as the BLE technology underlying the suggested OTP beacon system draws very little power and can reputedly operate stand-alone for 2 years on a very small battery[11].

As Smith[8] already suggests in his article on The Register, it is very straightforward to build an iBeacon compatible device from cheap off-the-shelf components, using a Raspberry Pi computer (which costs ±$30) and a simple Bluetooth 4.0 dongle (costing less than $10). To demonstrate the system proposed in this paper, the implementation of a demonstrator as an extension to the open source "tiqr" authentication system described in [4] is underway.

4.4 Applications

The related works discussed in Section 2 already suggest a number of use cases where using location as an authentication factor adds value. Of course, in controlled environments there are other means to enforce locality as an access condition to an application, for instance by limiting access to certain applications to specific network segments in specific office locations. This is a common approach, where the user's IP address is a deciding factor in whether or not they are allowed in to a system. Abstractly, this can also be seen as using location as an authentication factor.

Where a system such as the one proposed in this paper can, for instance, add real value is in environments with a shared infrastructure. Imagine, for instance, an office shared by many small companies, with a shared LAN. A company could place a beacon in its office such that employees can only access the company's servers if they are near the beacon (i.e. in the office). This may be a much cheaper solution than having to set up and manage something like a VPN.

Another area where the proposed system can offer a real added value is in authentication to cloud services. Businesses are increasingly moving critical applications to the cloud. On the one hand this saves cost, on the other hand this also means that authentication becomes critical to keeping confidential data safe. Again, in many modern office setups infrastructure may be shared, and workers are increasingly mobile (and use mobile devices), making it hard to enforce access the traditional way by granting access based on the IP address of the client. Using the system proposed in this paper solves that problem, providing a cheap, simple means to enforce location-based access control. The integration effort with cloud services is also small, since many cloud service providers already support multi-factor authentication schemes based on e.g. smartphone apps[12].

4.5 Comparison to other proximity-based technologies

Instead of using BLE, one could consider using other proximity-based technologies, such as "classic" Bluetooth or Near Field Communication (NFC). For completeness, a comparison of BLE with these technologies is given below:

4.5.1 "Classic" Bluetooth

As mentioned in Section 3.1, BLE is a completely new technology that, despite its name, differs significantly from classic Bluetooth. The biggest and most relevant differences - when considering the system proposed in this paper - are: operating distance, latency and mandatory pair-bonding. Classic Bluetooth has a much larger operating distance, of over 100m (of course, just as in BLE, this distance can be influenced by varying the signal strength of the radio). Where classic Bluetooth compares significantly less favorably when compared to BLE is connection setup latency; this is in the order of magnitude of 100ms compared to less than 3ms for BLE[13]. Combined with the fact that classic Bluetooth requires mandatory pairing of devices to bond them before any data can be exchanged this makes classic Bluetooth unsuitable for the type of application proposed in this paper.

4.5.2 Near Field Communication

Near Field Communication (NFC) is another proximity technology that has garnered a lot of attention over the past couple of years. NFC is often used in security applications, specifically in contactless smart cards. In principle, NFC could also be used to create a system such as the one proposed in this paper. The implementation and operation would be different though, for two reasons. Firstly, just like classic Bluetooth, NFC requires a connection to be established before any data can be exchanged[14]. Secondly, and more importantly, the operating distance of NFC is orders of magnitude lower than BLE. The maximum quoted operating distance is about 10cm (although in practice, more than 4cm can already be problematic). This drastically influences the usability compared to the BLE-based system proposed in this paper: users would need to actively "touch" an NFC beacon whereas the BLE-based system proposed here merely requires users to be e.g. in the same room as the beacon.

[11] Note that this is also true for other time-based OTP tokens.

[12] Consider for example Google Authenticator, a time-based OTP app by Google (`https://code.google.com/p/google-authenticator/`).

[13] Source: `http://www.bluetooth.com/Pages/low-energy-tech-info.aspx`

[14] Although it may be possible to piggyback broadcasting of an OTP on top of the transmission of the NFC device's ID field.

5. CONCLUSIONS AND FUTURE WORK

This paper demonstrates that it is possible to construct a simple, cheap to implement multi-factor authentication system that includes location as an extra factor in access control decisions. The advantage of the system outlined in this paper over other proposed systems is that it is very simple to implement, can be based on existing, unmodified open standards for OTP-based authentication and does not require complex calculations regarding the position of the user or prior knowledge of the position of radio aerials. Finally, the system is user-friendly, as it is invisible to the end user (they have to be authenticating from an authorized location equipped with a beacon, of course).

The solution proposed in this paper can add value in terms of security by adding an extra authentication factor – *somewhere the user is* – when it is incorporated into existing smartphone-based two-factor authentication systems.

To further expand upon the idea presented in this paper, there are a number of variants to the system that are worth further exploration in the future:

- Using a similar setup to replace time-based OTP tokens with a display on the token; in this variant, the OTP would be transmitted by a portable beacon rather than the user having to read it off the display. Since there is no protection on the channel between the beacon and the receiving device this setup will still require the beacon to be combined with another factor to make the setup secure.
- Adapting the system to be used for physical access control; a beacon with a relatively weak signal could be placed near a door. An OTP transmitted by this beacon could then be combined with a secret on e.g. a person's smartphone to authenticate to a remote server that could then signal the lock that incorporates the beacon to open.

6. ACKNOWLEDGEMENTS

The author would like to thank the Safet Acifović and Jorrit de Boer for their initial research into the iBeacon technology and its possible security applications as part of their coursework. Tony Smith's article on building your own iBeacon in The Register was also very helpful in validating the viability of the approach described in this paper. Finally the author would like to thank Erik Poll and Gergely Alpár for reviewing draft versions of this paper.

7. REFERENCES

[1] D. M'Raihi, J. Rydell, S. Bajaj, S. Machani, and D. Naccache. RFC 6287 - OCRA: OATH Challenge-Response Algorithm, 2011.

[2] Anind K. Dey. Understanding and Using Context. *Personal and Ubiquitous Computing*, 5(1):4–7, February 2001.

[3] Elisa Bertino and Michael Kirkpatrick. Location-Aware Authentication and Access Control Concepts and Issues. In *2009 International Conference on Advanced Information Networking and Applications*, pages 10–15. IEEE, 2009.

[4] Roland M. van Rijswijk and Joost van Dijk. tiqr : a novel take on two-factor authentication. In *Proceedings of LISA '11: 25th Large Installation System Administration Conference*, pages 81–97, Boston, MA, 2011. USENIX Association.

[5] Feng Zhang, Aron Kondoro, and Sead Muftic. Location-Based Authentication and Authorization Using Smart Phones. In *2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications*, pages 1285–1292. IEEE, June 2012.

[6] Wen-Bin Hsieh and Jenq-Shiou Leu. Design of a time and location based One-Time Password authentication scheme. In *2011 7th International Wireless Communications and Mobile Computing Conference*, pages 201–206. IEEE, July 2011.

[7] Wayne Jansen and Vlad Korolev. A Location-Based Mechanism for Mobile Device Security. In *2009 WRI World Congress on Computer Science and Information Engineering*, pages 99–104. IEEE, 2009.

[8] Tim Kindberg, Kan Zhang, and N. Shankar. Context authentication using constrained channels. In *Proceedings Fourth IEEE Workshop on Mobile Computing Systems and Applications*, pages 14–21. IEEE Comput. Soc, 2002.

[9] YounSun Cho, Lichun Bao, and Michael T. Goodrich. LAAC: A Location-Aware Access Control Protocol. In *2006 Third Annual International Conference on Mobile and Ubiquitous Systems: Networking & Services*, pages 1–7. IEEE, July 2006.

[10] D. M'Raihi, S. Machani, M. Pei, and J. Rydell. RFC 6238 - TOTP: Time-based One-Time Password Algorithm, 2011.

[11] D. M'Raihi, M. Bellare, F. Hoornaert, D. Naccache, and O. Ranen. RFC 4226 - HOTP: An HMAC-Based One-Time Password Algorithm, 2005.