POSTER: Quasi-ID: In fact, I am a human.

Milivoj Simeonovski
CISPA, Saarland University
Saarbrücken, Germany
simeonovski@cs.uni-saarland.de

ABSTRACT

CAPTCHAs (Completely Automated Public Turing test to tell Computers and Humans Apart) are the dominantly used turing tests to protect websites against bots that are impersonating human users to gain access to various types of services. The test is designed in a way to be very difficult for robotic programs, but comfortably easy for humans. As artificial intelligence research thrives towards the biggest challenge of the field — simulating the work of a human brain — the complexity of CAPTCHA tests increases, making it more and more difficult for humans to answer the tests. The problem gets even bigger, with the latest research reports in fact indicating that CAPTCHAs are broken.

We present Quasi-ID: a novel approach for determining whether or not a user is a human in a scalable and privacy-preserving manner. Our system utilizes smart devices as ubiquitous input devices for invoking a physical interaction with the users. Such an interaction between the user and his smart device can prove that the user is actually a human.

Support for Quasi-ID can be deployed today along with the current CAPTCHA solutions. It does not add additional burden to the web service and requires a non-persistent communication with the Quasi-ID service provider.

Categories and Subject Descriptors

H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*Input devices and strategies.*; D.4.6 [**Security and Protection**]: Access control and authentication

Keywords

CAPTCHA; two-step verification; human factors; authentication ticket; pseudonyms; privacy; unlinkability

1. INTRODUCTION

To prevent abusive behavior by automated programs (often referred to as bots), web services deploy certain kinds of challenges to determine whether a user is human or a computer program. These challenges usually comprise a distorted series of letters or objects presented in a way that a user should not have any difficulties to solve the puzzle and recognize the text or the object within the picture. At the same time, the distorted object should not be recognizable for computers. Such a challenge that is prevalently deployed today by many web services is called a CAPTCHA [6, 7] (Completely Automated Public Turing test to tell Computers and Humans Apart). CAPTCHAs are used to protect web services against various type of attacks, such as password brute-force attacks, and to reduce spam.

As the research in the field of artificial intelligence (AI) gets closer to simulating the human brain, it also gets easier for computer algorithms to solve the state-of-the-art CAPTCHA puzzles. To mitigate this problem, developers continuously increase the distortion level of CAPTCHAs making these challenges not only difficult for machines but for humans as well. Challenges with an increased distortion level make it difficult for people to get the right answer directly at their first attempt. Thus, it can happen that they require several attempts trying to solve the challenge, which is frustrating for the users. In addition to that, there is evidence [5] that latest research on AI has not only broken simple CAPTCHAs but even more sophisticated ones that are used by many of the services today.

In search for a better solution. Determining whether or not a user is human is a challenging problem which involves a number of research disciplines including security, usability and AI. We do not aim to replace CAPTCHAs, but rather promoting a new different approach for solving this problem. The approach we propose is based on three key insights. First, the key problem with the current way of proving that a user is a human is the way how the (CAPTCHA) turing tests are designed. Today, all approaches are based on a simple image recognition or reconstruction. The AI research has already reached the required level of intelligence that beats traditional CAPTCHAs. Our second key insight is the ubiquity of smart devices. Statistics show that the majority of the internet users have at least one smart device. If, for instance, the smart device is connected to the user's identity, that can be a good enough proof that the owner of the phone is actually a human. This insight, has been successfully used by some services to add an extra layer of security, and is usually referred to as two-factor authentication. Our goal is to apply this mapping (user → smart device) in a privacy preserving manner, *i.e.,* without exposing the user's real identity. Moreover, as an addition to privacy, we do not want to allow any single party to be able to link two

CCS'14, November 3–7, 2014, Scottsdale, Arizona, USA.
ACM 978-1-4503-2957-6/14/11.
http://dx.doi.org/10.1145/2660267.2662378.

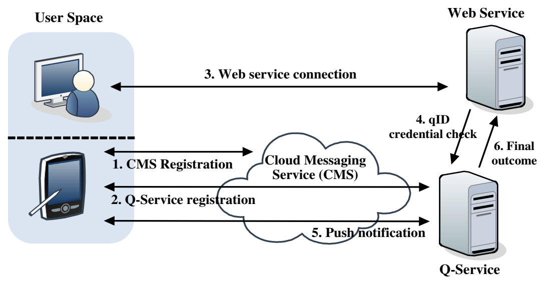

Figure 1: The Quasi-ID system architecture

or more user requests to a single identity, *i.e., unlinkability.* Finally, our third key insight is the fact that employing the current turing challenges makes the communication between the users and the services more difficult. Instead of simplifying the communication, the current widely deployed solutions add an additional burden for the users by having to solve a problem not even related to the communication.

Our contribution. In this work, we propose a different approach for proving that a user is actually a human. Based on the above three insights, we design a framework such that a web service can get a valid confirmation that a requesting user is not a robot, and the user can convince the service that he is a human by a simple interaction with his smart device. All this can be done in an efficient, privacy preserving, unlinkable, and less frustrating manner.

Our framework utilizes ubiquitous smart devices as input devices for establishing an additional communication channel between the web service and the user via the user's smart device. Isolating such a channel, enables the service to directly challenge the user to physically interact with his smart device, and thus prove that he is a human.

Technically, the framework relies on the user's physical interaction associated to a pseudonymous identity and subsequently uses temporary pseudonym signatures [1] to let the user sign a service request representing his eligibility to use the system.

The design of the framework ensures that a physical interaction between the user and his smart device has been performed. Once such an interaction is performed, the Quasi-ID service provider issues a confirmation to the web service that the requesting user is indeed human. Specifically, we propose a new primitive called Quasi-ID ticket (qID), a time bounded pseudonymous ticket, that (1) gives a user access to the Quasi-ID system and (2) triggers the user (via his smart device) to actively perform a physical interaction.

2. DESIGN OVERVIEW

In total, our framework defines four parties (Figure 1), namely, the web service (WS), the Quasi-ID service provider (Q-Service), the cloud messaging service (CMS), and the user (U) who is in a possession of a smart device. The basic scenario we are concerned with in this paper would be as follows. The user contacts the WS in order to benefit from some of the services it provides. The WS wants to protect its resources by deploying a mechanism that gives users the possibility to prove that they are humans. The Q-Service is a service provider responsible for contacting the user in order to prove that he is a human, and finally, the CMS acts as a communication channel between the Q-Service and the user. In the following, we describe our system goals and trust assumptions. Then, we present our key idea.

Goals. The framework is mainly intended to provide protection for web services against automated programs by determining whether or not the requesting user is a human. Our framework should be *efficient* for both, the users and the web services, and should also be easy to *deploy* for the web services. Additionally, the framework should preserve the user's *privacy* and should not let any third party, not even the service provider, to be able to link two or more user requests. Therefore, it should provide *unlinkability.*

Trust assumptions. We do not trust the web service and we want to ensure that the service does not benefit from any additional information about the requesting user except that he is a human. Naturally, we allow the user space to be corrupted and controlled by an attacker. Not trusting the user space is important because a corrupted user is a realistic threat. The Quasi-ID service provider is honest, *i.e.,* it does not lie about the user's behavior. And finally, we assume that the framework employs a mechanism that can prove a physical interaction between the user and his smart device. For instance, one solution could utilize the Quire framework [2] to reliably distinguish legitimate interactions from forgeries.

Key Idea. The key idea of Quasi-ID is to use temporal pseudonymous identities (α) for the users, and temporary authentication tickets (qID), such that for every qID verification a new physical interaction between the requesting user and his smart device is required. We define qID to be a tuple of the form

$$qID = \langle \alpha, t, \sigma_\alpha \rangle \tag{1}$$

where $\alpha = g^x$ is a pseudonym generated by the user, such that g is a generator of a cyclic group $\mathbb{G}$ of prime order p with the security parameter k and $x \in_R Z_p$ is a random secret value known only to the user. The second parameter is a current timestamp, and the last parameter σ_α is a signature over α and t generated by the service provider (Q-Service). This signature proves that a user with identity α is registered with the Q-Service.

We also employ the concept of *pseudonym signatures* where (x, g^x) is a signing key pair as defined in [1]. Pseudonyms are generated independently for every identity and it is not possible to link two or more pseudonyms to a single identity nor to identify the actual user behind that pseudonym. Our protocol is based on temporary identities, therefore, the pseudonym signature scheme is an ideal candidate for a signature scheme to be used. The user utilizes them to sign the service requests without being identified nor linked by the verifier.

3. QUASI-ID DESIGN

Figure 1 presents a general expected architecture to achieve the above mentioned goals. We assume that the web service is already registered with the Q-Service and has its own credentials. In the following, we describe the steps of the protocol (these steps are also illustrated in Figure 1). We define all the steps as two-party protocols of type User-Service ($U \leftrightarrow S$) and Service-Service ($S \leftrightarrow S$).

1. ($U \leftrightarrow CMS$): Figure 1 step 1

The user runs the CMS registration module. This module is about registering the user's smart device within the CMS such as GCM (Google Cloud Messaging) [4]. The user provides the ID of the Q-Service, and gets a personal registration number α_c as an answer. This number is issued by the CMS servers to the application on the smart device that allows it to receive messages. In other words, α_c is tied to a particular application running on a particular device.

2. ($U \leftrightarrow$ Q-Service): Figure 1 step 2

   Upon receiving α_c, the user starts a key exchange protocol (1W-AKE[1]) with the Q-Service. He generates a new random pseudonym α, and sends it along with α_c to the Q-Service following the protocol.

   The Q-Service uses α_c to identify each device that has registered with the Quasi-ID protocol to receive messages from the Q-Service. The Q-Service generates the qID ticket as defined in (1) and sends it back to the user. Note that the communication between the user and the Q-Service is indirect, *i.e.,* it runs through the CMS cloud, and is encrypted.

3. ($U \leftrightarrow WS$): Figure 1 step 3

   After the initial registration, the user is ready to establish a TCP connection to a web service. First, he creates a signature $\sigma_{qID} = H(qID||service)^x$ for the ticket qID, where H is a secure hash function, $service$ is the destination service, and x is the secret corresponding to his pseudonym $\alpha = g^x$. Then, along with a standard http request, U sends his qID ticket and created σ_{qID} signature the to the destination server[2].

4. ($WS \rightarrow$ Q-Service): Figure 1 step 4

   Once the signed request reaches the WS, it first verifies the signature σ_α from the qID ticket, using the public key of Q-Service. After a successful verification, it sends the qID along with σ_{qID} to the Q-Service asking for a confirmation that the user owning the qID is a human.

5. (Q-Service $\leftrightarrow U$ via CMS): Figure 1 step 5

   Upon a successful signature verification of σ_{qID}, Q-Service sends a push message to U requesting a proof of a physical interaction. Note that our framework is generic in nature, and does not define the way how the user physically interacts with his smart device.

6. (Q-Service $\rightarrow WS$): Figure 1 step 6

   Finally, the Q-Service informs the web service about the final outcome.

After completion of the protocol, the pseudonymous identity α and the authentication ticket qID are discarded. For any future service requests, a new protocol execution is required. We achieve the desired privacy property by using an intermediate cloud messaging service for a communication between the user and the service provider. To achieve the unlinkability property, we require a pseudonym renewal for every protocol execution. Nevertheless, a collaboration between CMS and the Q-Service breaks the unlinkability property.

[1]One-way authenticated key exchange protocol [3] that provides one-way anonymity.
[2]The qID information will be sent as a part of the URL string.

We provide an experimental demo implementation to show the practicality of our framework. The implementation is based on the GCM services [4] for establishing a communication channel between the Q-Service and the user's smart devices. Finally, we can conclude that such a realization does not add additional burden to the web service and merely requires a non-persistent communication with the Quasi-ID service provider. Therefore, support for Quasi-ID can be deployed today along with the current CAPTCHA solutions.

4. DISCUSSION AND FUTURE WORK

In fact, the latest evidence of broken CAPTCHAs places the current technology in a very uncertain position. However, we believe that replacing CAPTCHAs with a different approach is not a question of giving up one dominant technology for something unarguably better, but of giving up one set of compromises in exchange for another.

In this work, we presented Quasi-ID: a novel approach for determining whether or not a user is a human. Our approach utilizes smart devices as ubiquitous input devices for invoking a physical interaction with the users. The framework thereby preserves the privacy of the requesting user as well as it provides unlinkability, *i.e.,* users cannot be linked by the Quasi-ID service provider or by the web service.

For future work, we plan to make a user study about the usability of the proposed framework and formalize the corresponding security definitions, privacy and unlinkability.

5. REFERENCES

[1] M. Backes, J. Clark, A. Kate, M. Simeonovski, and P. Druschel. Backref: Accountability in anonymous communication networks. In *ACNS'14*.

[2] M. Dietz, S. Shekhar, Y. Pisetsky, A. Shu, and D. S. Wallach. Quire: Lightweight provenance for smart phone operating systems. In *USENIX Security Symposium*, 2011.

[3] I. Goldberg, D. Stebila, and B. Ustaoglu. Anonymity and one-way authentication in key exchange protocols. *Des. Codes Cryptography.*

[4] Google Inc. Google cloud messaging. Online: http://developer.android.com/google/gcm/index.html. [Accessed July-2014].

[5] Vicarious. Vicarious ai passes first turing test: Captcha. Online: http://news.vicarious.com/post/65316134613/vicarious-ai-passes-first-turing-test-captcha. [Accessed July-2014].

[6] L. von Ahn, M. Blum, N. J. Hopper, and J. Langford. Captcha: Using hard ai problems for security. In *EUROCRYPT*, pages 294–311, 2003.

[7] L. Von Ahn, B. Maurer, C. McMillen, D. Abraham, and M. Blum. recaptcha: Human-based character recognition via web security measures. *Science*, 321(5895):1465–1468, 2008.