Continuous and Transparent Authentication

A secure and passwords-less experience using wearables

Jacob Benjamin Cholewa

Department of Computer Science
IT University of Copenhagen

Spring, 2017

Outline

What is authentication?

What is the purpose of authentication?

What is authentication?

Authentication denotes the process by which the identity of a user (or any subject) is verified. In this process, the user provides his claimed identity together with evidence in the form of credentials. If the authenticating system accepts the evidence, the user is successfully authenticated.

What is authentication?

Passwords are bad!

In 1979 Morris and Thompson showed how they were able to crack 81% of 3000 user generated passwords, using only a small dictionary.

What is authentication?

Devices are everywhere

Authentication is simply not practical. Imagine that you would have to type your username and password at every pervasively available device before being able to use them.

What is authentication?

Devices are everywhere

"Clearly, if the pattern of login and logout is not considered a usability problem today, it will most certainly become one in the years to come." (Bardram et. al 2003).

Authentication Properties

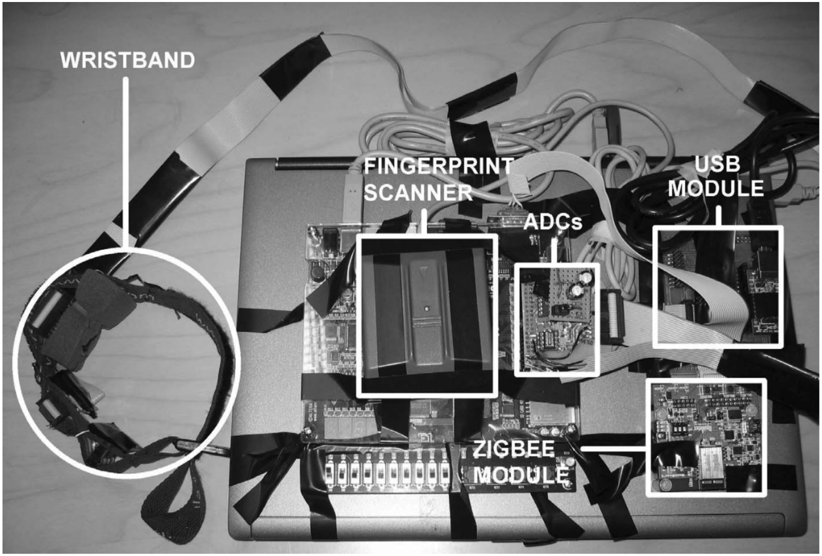
Usability								Deployability						Security										
Memorywise-Effortless	*Scalable-for-Users*	*Nothing-to-Carry*	*Physically-Effortless*	*Easy-to-Learn*	*Efficient-to-Use*	*Infrequent-Errors*	*Easy-Recovery-from-Loss*	*Accessible*	*Negligible-Cost-per-User*	*Server-Compatible*	*Browser-Compatible*	*Mature*	*Non-Proprietary*	*Resilient-to-Physical-Observation*	*Resilient-to-Targeted-Impersonation*	*Resilient-to-Throttled-Guessing*	*Resilient-to-Unthrottled-Guessing*	*Resilient-to-Internal-Observation*	*Resilient-to-Leaks-from-Other-Verifiers*	*Resilient-to-Phishing*	*Resilient-to-Theft*	*No-Trusted-Third-Party*	*Requiring-Explicit-Consent*	*Unlinkable*

Authentication Properties

				Usability								Deployability						Security										
Category	Scheme	Described in section	Reference	*Memorywise-Effortless*	*Scalable-for-Users*	*Nothing-to-Carry*	*Physically-Effortless*	*Easy-to-Learn*	*Efficient-to-Use*	*Infrequent-Errors*	*Easy-Recovery-from-Loss*	*Accessible*	*Negligible-Cost-per-User*	*Server-Compatible*	*Browser-Compatible*	*Mature*	*Non-Proprietary*	*Resilient-to-Physical-Observation*	*Resilient-to-Targeted-Impersonation*	*Resilient-to-Throttled-Guessing*	*Resilient-to-Unthrottled-Guessing*	*Resilient-to-Internal-Observation*	*Resilient-to-Leaks-from-Other-Verifiers*	*Resilient-to-Phishing*	*Resilient-to-Theft*	*No-Trusted-Third-Party*	*Requiring-Explicit-Consent*	*Unlinkable*
(Incumbent)	Web passwords	III	[13]			●		●	●	○	●	●	●	●	●	●	●		○						●	●	●	●
Password managers	Firefox	IV-A	[22]	○ 🟩	● 🟩	○ 🟥	○ 🟩	●	●	● 🟩	🟥	●	●	●	🟥	●	●	○ 🟩	○					● 🟩	●	●	●	●
	LastPass		[42]	○ 🟩	● 🟩	○ 🟥	○ 🟩	●	●	● 🟩	○ 🟥	●	○ 🟥	●	🟥	●	🟥	○ 🟩	○	○ 🟩	○ 🟩		○ 🟩	● 🟩	●	🟥	●	●

Motivation

State of the art (almost)



Research Questions

Existing research shows that wearable devices offer a both inexpensive and simple way of enabling Continuous and Transparent Authentication (CTA). However, work is still needed in either building or adapting a suitable security protocols as well as making CTA possible with commodity hardware and services.

Therefore this thesis aims to:

1. Put forward a requirement analysis and design of a CTA scheme.
2. Either select or design a suitable protocol for CTA.
3. Show that the protocol can be used in a commodity hardware implementation of the proposed scheme.

Scenarios

Establishing a session

A client wants to establish a session with a service provider. The service provider replies with a challenge. The client now asks the authenticator to solve the challenge. The authenticator, with optional end-user consent, and optionally in collaboration with its sibling, solves the challenge. When presented with the correct answer the service provider authorizes the client. The session is maintained as long as the authenticator remains unlocked and the authenticator remains in proximity to the client.

Outline

ElGamal encryption

Cyclic groups

Let $\mathbb{G}$ be a set of order (size) $m = |\mathbb{G}|$. We call g a generator of $\mathbb{G}$ if all values of $\mathbb{G}$ can be expressed as $\{g^1, g^2 \cdots g^m\}$.

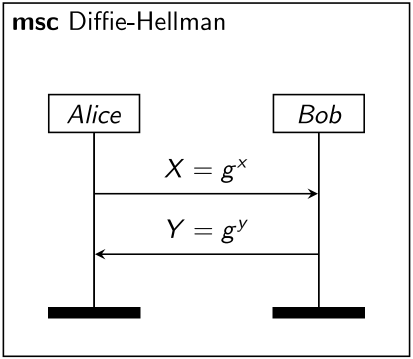
These groups are heavily used in cryptography because for certain groups the discrete logarithm is difficult. $(log_g(g^x) = x)$

ElGamal encryption

Recall Deffie-Hellman

The Diffie-Hellman key exchange protocol allows us to compute a shared key in the open without leaking any secrets.

$$key = (g^x)^y \equiv (g^y)^x$$



ElGamal encryption

Let g be a generator of group $\mathbb{G}_p$. Select a private key $x \in \mathbb{Z}_p$, and a corresponding public key $y = g^x$. We can then encrypt a message m as

$$enc(m, y) \to c = (\alpha, \beta) = (my^r, g^r)$$

where r is an arbitrary number in $\mathbb{Z}_p$. A message can then be decrypted by computing

$$dec(c, x) := \frac{\alpha}{\beta^x} = \frac{my^r}{(g^r)^x} = \frac{m\cancel{(g^x)^r}}{\cancel{(g^r)^x}} = m$$

ElGamal encryption

Homomorphism

For any asymmetric encryption scheme $\Pi = (Enc, Dec)$ and key-pair (x, y), where it holds that $Dec(Enc(m, x), y) = m$, then the decryption primitive is homomorphic over the keys if[1]

$$Dec(c, x_1) \times Dec(c, x_2) \equiv Dec(c, x_1 + x_2)$$

This can be used to make distributed decryption where each party partially decrypts the ciphertext and combines the shares to recover the message.

$$\prod_{i=1}^{n} Dec(c, x_i) = m$$

(under the assumption that the message was encrypted with a shared public key corresponding to $\sum_{i=1}^{n} x_i$)

[1] For any set of operations '$+$' and '$\times$'

ElGamal encryption

Distributed decryption

Recall the exponentiation rule $a^x \cdot a^y = a^{x+y}$.

We can use this to make distributed ElGamal. For a group of n agents we can distribute the private key into n shares such that the private and public key is:

$$x = \sum_{i=1}^{n} x_i \qquad y = \prod_{i=1}^{n} y_i$$

ElGamal encryption

Distributed decryption

With the distributed private key we can make partial encrypting by having each party calculate β^{x_i}

$$m = \frac{\alpha}{\prod_{i=1}^{n} \beta^{x_i}}$$

Show the sequence diagram

Zero-Knowledge Proofs

Interactive proof

In cryptography, a zero-knowledge proof or zero-knowledge protocol is a method by which one party (the prover) can prove to another party (the verifier) that a given statement is true, without conveying any information apart from the fact that the statement is indeed true.

Zero-Knowledge Proofs

Interactive proof

This is a classic Σ-protocol by Schnorr [Sch91]. Alice and Bob know v and g, but only Alice knows x, so that $v = g^x$.

1. Alice chooses z at random and sends $a = g^z$ to Bob.
2. Bob chooses a challenge c at random and sends it to Alice.
3. Alice sends $r = (z + cx) \mod q$ to Bob.
4. Bob checks that $g^r = av^c$.

Zero-Knowledge Proofs

Non-interactive proof

Using the Fiat-Shamir heuristic we can make the proof non-interactive using hash functions.

1. Alice chooses z at random and sends $a = g^z$ to Bob.
2. Alice computes $c = H(g, y, z)$
3. Alice sends $r = (z + cx) \mod q$ to Bob.
4. Bob computes c and checks that $g^r = av^c$.