Security Protocols and Application — Final Exam Part 1/2

Solution

Philippe Oechslin and Serge Vaudenay

19.6.2012

- duration: 2h00
- no document allowed
- a pocket calculator is allowed
- communication devices are not allowed
- the exam invigilators will not answer any technical question during the exam
- the answers to each exercise must be provided on separate sheets
- readability and style of writing will be part of the grade
- do not forget to put your name on every sheet!

Google Authenticator

Q.1 Google Authenticator provides *strong* authentication. What does *strong* refers to in this case?

> *This is a 2-step authentication. Authentication requires a password ("what you know") and a verification code which is compute by a personal device ("what you have").*

Q.2 Assume a browser-based application using Google Authenticator. When prompted, how would you get the verification code?

> *It can be received by SMS, voice mail, or computed in an offline way by a specific app on a smart phone.*

Q.3 For browser-based applications using Google Authenticator, what does "remember verification for this computer" mean?

> *It stores a cookie and don't bother again with authentication in the next 30 days as long as the cookie is provided. To use this, the computer must be trusted.*

Q.4 In Google Authenticator, how long (how many bits) is the shared secret which generates the verification codes, and how is it set up?

> *It is loaded with a QR code or with a 16-character base32 string. So, it has 80 bits.*

Q.5 What is the impact of finding collisions on SHA-1 on the security of Google Authenticator?

> *Nothing. SHA-1 is used as a pseudo-random function to generate verification codes.*

Q.6 If an adversary tries verification codes at random, how many attempts does he need before succeeding?

The verification code is of 6 decimal digits, so the adversary needs 10^6 *attempts on average. That is, about* 2^{20}.

Q.7 What is the protection against automated verification code guessing attacks?

After 3 invalid codes, there is a Captcha. In theory, robots cannot answer to captchas automatically.

Q.8 What is the difference between HOTP and TOTP? Why do we prefer one to the other?

HOTP is based on a counter. The HOTP credential remains valid until it is used. So, if it is stolen, the adversary has plenty of time to use it.

TOTP is based on absolute time. So, TOTP credential expires fast enough. This is why TOTP is preferred.

Q.9 In Google Authenticator, an algorithm uses HMAC on a clock-based value to compute the verification code. How is this clock-based value calculated?

It is the number of 30” *periods until epoch.*

Q.10 How much time is a TOTP verification code valid in Google Authenticator?

It is valid for 4 minutes (8 30” *periods), or until used. A verification code cannot be used twice.*

Q.11 How can we continue to use Google Authenticator if the smart phone computing the verification code is lost or broken?

We must have a set of backup codes. These codes are valid until used or a new set of 10 backup codes is generated.

Q.12 How to use Google Authenticator for non browser-based applications? What is the advantage compared to authentication without Google Authenticator?

We have to generate an application-specific password. It is not a 2-step authentication but one advantage is that this specific password can be revoked. It is better than a single password for all applications.

Q.13 Describe a man-in-the-middle attack for a browser-based application using Google Authenticator. How to defeat it?

A malicious browser-based application can simulate the Google Authenticator box to get the password and verification code, then use it to impersonate the victim on another browser-based application.

To defeat this attack, the browser-based application must be authenticated (e.g. based on a certificate and TLS connection) and trusted.