# Security Tests for Mobile Applications - Why using TLS/SSL is not enough.

Peter Kieseberg, *Member, IEEE,* Peter Frühwirt, Sebastian Schrittwieser, and Edgar Weippl, *Member, IEEE*

***Abstract*—Security testing is a fundamental aspect in many common practices in the field of software testing. Still, the used standard security protocols are typically not questioned and not further analyzed in the testing scenarios. In this work we show that due to this practice, essential potential threats are not detected throughout the testing phase and the quality assurance process. We put our focus mainly on two fundamental problems in the area of security: The definition of the correct attacker model, as well as trusting the client when applying cryptographic algorithms.**

***Keywords*—*Testing, Security, TLS/SSL***

## I. Motivation and Related Work

SECURITY issues in productive software are nowadays omnipresent. Still, many of these issues are not based on new scientific findings or fundamental research, but on the exploitation of known weaknesses, very often either caused by ignorance with respect to possible attack vectors, or by relying on security mechanisms that are unsuitable for the given task or architecture. One very prominent example is the TLS protocol [1], [2] (Transport Layer Protocol), a successor of the original SSL protocol [3]. TLS is currently one of the most popular security protocols and is accepted as a secure means for secret communication (e.g. [4] for SSL). However, TLS can only guarantee the protection of the communication to the respective end point, but not the security of the underlying protocols that e.g. implement the authentication of users with services. In case these underlying mechanisms are affected by design or implementation errors, it is possible to exploit this in the course of attacks, even in case of working encryption (e.g. [5], [6]). Additionally, another fundamental issue is often neglected: The sole purpose of TLS lies in the protection of communication between two *trustworthy end points*, i.e. both communication partners know and trust each other. Still, in many applications this prerequisite is not applicable, especially considering mobile applications, where the end user has to be taken into account as potential attacker (see [5], [7]). In this work we will discuss some attack vectors against popular and well distributed software products that yield one of these two problems that could be mitigated by a suitable testing strategy.

P. Kieseberg, P. Frühwirt and E. Weippl are with SBA Research, Vienna, Austria, ([firstletterfirstname][lastname])@sba-research.org).

S. Schrittwieser is with the University of Applied Scioences, St. Pölten, Austria, ([firstname].[lastname]@fhstp.at).

2015 IEEE Eighth International Conference on Software Testing, Verification and Validation Workshops (ICSTW)
13th User Symposium on Software Quality, Test and Innovation (ASQT 2015)
978-1-4799-1885-0/15/$31.00 

## II. The testing approach

Developers often rely on the assumption that they possess full control over their client software, i.e. the smartphone application, thus following that it is perfectly safe to trust these applications when they contact the respective servers through the Internet. This approach can lead to serious security issues, since manipulation of data may still be possible on the transport layer, even though transport layer encryption like SSL/TLS is chosen to protect weak protocols. Furthermore, this additional encryption layer adds even more complexity to the testing phase and sometimes even prohibits extensive security testing. This section covers the fundamental approach, how to circumvent the encryption between the smartphone application and the server in order to analyze (and test) the protected protocols for weaknesses in design and/or implementation. This approach relies on the fact that, contrary to the implicit assumption of the developer, the client is under the full control of the tester (or attacker), since he/she can manipulate hard- and software, as well as the operating system. The communication between the two end points is encrypted and secured with a certificate, i.e. an attacker is not able to intercept the communication via a so-called man-in-the-middle attack without making this detectable by the client through the falsified certificate. Still, due to the client being under the control of the tester/attacker, it is possible to trust the certificate system-wide, i.e. the system settings of the smartphone can be changed in order to always trust this (false) certificate, thus enabling to route all traffic of the client through a proxy. Contrary to the original setup, the TLS encryption is now not established between the client and the server, but the communication is split on the transport layer into two separate communication paths, one between the client and the proxy and the other between the proxy and the server. While both paths are still protected by the TLS encryption, the proxy is now able to read the communication. Figure II shows the basic implementation of this testing proxy for enabling protocol tests in encrypted environments. The tester using the proxy in-between the server and the client is not only able to (passively) read the communication, but he/she can also arbitrarily alter or even block the communication.

## III. Case Study

In this section we demonstrate the consequences of the described testing weaknesses that can be found by utilizing the strategy outlined in Section II by means of real examples. We chose applications with a significant user base, covering several fields of application.

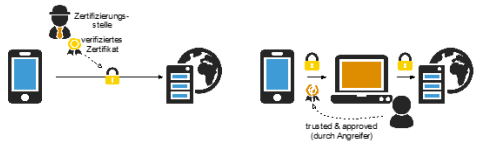


Fig. 1. Man-in-the-middle for protocol analysis

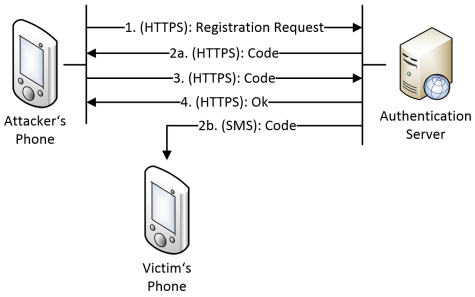


Fig. 2. Exploiting the Wow-Talk authentication.

*a) Mobile Instant Messengers:* This class of mobile Apps has been analyzed frequently during the last few years, especially regarding the authentication methods applied (e.g. [5]), which is one of the most vital parts regarding security in these applications. While often relying heavily on TLS, most applications incorporate some form of two-factor authentication, typically via SMS. Still, many of them include vital security flaws as outlined by [5] and [7]. One prime example can be seen in Figure III-0a, the authentication mechanism in WowTalk. The user (legit or fraud) enters a request for authentication in the app which triggers the sending of a SMS to this number containing a short code which needs to be entered in the app in order to verify the possession of the claimed number. Still, the number is also sent to the device via HTTPS (TLS), which can be intercepted with the approach outlined in Section II. Blackbox testing will not detect this design error due to the protection by TLS.

*b) Games and Entertainment:* They may not strike as vital applications, still, a lot of money is involved in this industry and techniques targeting the right software (e.g. online gambling) may even introduce damage scenarios against other users. For our demonstration the well-known game "Curiosity" was chosen, where a very large cube consisting of small elements had to be mined down. This could be advanced by purchasing better "mining tools" and bonuses. Still, these bonuses were set by the client and could thus be exploited.

*c) Social Networks:* We analyze the Photoswap social network which offers sending photos to other, previously unknown, users and getting other photos in return. In case the photo seems interesting, it is possible to reply and start a conversation by photo. Internally the application works by sending the link to a picture on the server via HTTPS, which is then downloaded and shown to the user. The main issue is that all the photos are stored with very predictable names - instead of using a hash-value based approach, the file names are constructed via a pattern. By bypassing the TLS layer it is possible to retrieve this pattern and to brute-force the names of all existing photos making them downloadable from external sources.

*d) Online Shops and Ticketing:* Also other applications, like e.g. Navigation systems which rely on the user buying maps or online shops and ticketing systems are often prone to these kinds of attacks, which could be mitigated by adapting the testing strategy accordingly.

## IV. Conclusion

In this work we were able to demonstrate that even widely used software products possess fundamental flaws in the design of their communication protocols, even though secure cryptographic standards were followed in order to achieve communication security. This is especially important for the field of software testing, since in the area of security a working testing strategy needs to question the security and especially applicability of said standards critically.

## Acknowledgment

This research was funded by COMET K1, FFG - Austrian Research Promotion Agency.

## References

[1] C. Allen and T. Dierks, "The tls protocol version 1.0," 1999.

[2] T. Dierks, "The transport layer security (tls) protocol version 1.2," 2008.

[3] K. Hickman and T. Elgamal, "The ssl protocol," *Netscape Communications Corp*, vol. 501, 1995.

[4] D. Wagner, B. Schneier *et al.*, "Analysis of the ssl 3.0 protocol," in *The Second USENIX Workshop on Electronic Commerce Proceedings*, 1996, pp. 29–40.

[5] S. Schrittwieser, P. Frühwirt, P. Kieseberg, M. Leithner, M. Mulazzani, M. Huber, and E. R. Weippl, "Guess who's texting you? evaluating the security of smartphone messaging applications." in *NDSS*, 2012.

[6] S. Schrittwieser, P. Frühwirt, P. Kieseberg, M. Leithner, M. Mulazzani, M. Huber, G. Wondracek, S. Rennert, and E. R. Weippl, "Secure software in der cloud." in *Cloud und Klein: IT im Spannungsfeld zwischen Servercluster und Sensornetz,*, 2012.

[7] R. Mueller, S. Schrittwieser, P. Fruehwirt, P. Kieseberg, and E. Weippl, "What's new with whatsapp & co.? revisiting the security of smartphone messaging applications," in *Proceedings of the 16th International Conference on Information Integration and Web-based Applications & Services*. ACM, 2014, pp. 142–151.

**Peter Kieseberg** received a masters degree in Technical Mathematics in Computer Science from the Vienna University of Technology. He worked as a consultant in the telecommunication sector for several years before joining SBA Research. His main research interests include digital forensics, cryptography and mobile security.

**Peter Frühwirt** received a bachelors degree in Software and Information Engineering and in Business Informatics, as well as a master's degree in Software Engineering from the Vienna University of Technology. In 2015 he obtained his PhD in Security for his work in the area of Database Forensics.

**Sebastian Schrittwieser** received his master's degree in Business Informatics with focus on IT security from the Vienna University of Technology and his PhD in Security for his work in the area of Software Obfuscation.

**Edgar Weippl** After graduating with a Ph.D. from the Vienna University of Technology, Edgar worked in a re-search startup for two years. He then spent one year teaching as an assistant professor at Beloit College, WI. In 2004 he joined the Vienna University of Technology and co-founded the research center SBA Research.