M.Sc. thesis project(s):
Testing of Lightweight Security Protocols

Background Within the project aSSIsT (`https://assist-project.github.io/`), we are developing techniques for ensuring that implementations of security protocols (as well as other security-critical software) conform to their specifications, and are free from security vulnerability. In particular, we care about software for security of IoT devices. A motivation is that the number of Internet of Things (IoT) devices is projected to reach 11.6 billion by 2021, thus constituting half of all devices connected to the Internet.

Security protocols are a key building block in IoT systems. Implementations of such network protocols must conform to their specifications in order to avoid security vulnerabilities and interoperability issues. Even seemingly innocent deviations from the standard specification may expose implementations to security attacks. We have previously developed infrastructure for testing Datagram Transport Layer Security (DTLS) [3, 4], which is one of the primary protocols for securing IoT applications [5]. We have applied and further developed several techniques for testing DTLS implementations. One is state fuzzing [2], others are symbolic execution (publication under submission).

We plan to apply our techniques also to other lightweight security protocols. One protocol under consideration is EDHOC, which is designed to be more lightweight than DTLS and more suitable for resource-constrained environments. EDHOC is in the final stages of standardization [1] (a primer is in `https://hal.inria.fr/hal-03434293`), and infrastructure for testing EDHOC will be of high interest as new implementations are under development. EDHOC reuses several concepts and techniques from TLS and DTLS. An M.Sc. project would therefore investigate how to retarget the efforts we have performed for DTLS, concerning either state fuzzing or symbolic execution, and apply it to (a fragment of) EDHOC.

Suggested Project(s) The project will develop infrastructure for testing impementations of the EDHOC protocol. The infrastructure should be able to

- send arbitrary EDHOC packets in arbitrary order to an implementation under test (SUT),
- modify selected fields in sent packets from a suitable API, and
- receive and parse arbitrary EDHOC packets.

It is very helpful to have an existing EHOC implementation as a starting point. Such an implementation should have methods for composing, sending, and parsing EDHOC messages. A possible such starting point is the Java implementation by Marco Tiloca `https://github.com/rikard-sics/californium/tree/edhoc-dev`, which should have methods for all these functionalities.

An inspiration for the project is a testing tool for TLS, TLS-Attacker [6], which is a tool for testing TLS implementations, with a rich set of functionalities.

Application Send your application for M.Sc. project to one of the contact persons below.

Supervisors and Contact persons Bengt Jonsson (`bengt@it.uu.se`), Kostis Sagonas (`kostis@it.uu.se`), Parosh Abdulla (`parosh@it.uu.se`), Paul Fiterau-Brostean (`paul.fiterau_brostean@it.uu.se`).

References

[1] 2021. https://datatracker.ietf.org/doc/draft-ietf-lake-edhoc.

[2] Paul Fiterau-Brostean, Bengt Jonsson, Robert Merget, Joeri de Ruiter, Konstantinos Sagonas, and Juraj Somorovsky. Analysis of DTLS implementations using protocol state fuzzing. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 2523–2540. USENIX Association, August 2020.

[3] Nagendra Modadugu and Eric Rescorla. The design and implementation of datagram TLS. In *Proceedings of the Network and Distributed System Security Symposium*, NDSS 2004, 2004.

[4] Eric Rescorla and Nagendra Modadugu. Datagram transport layer security version 1.2. Internet Requests for Comments, January 2012.

[5] Zach Shelby, Klaus Hartke, and Carsten Bormann. The constrained application protocol (CoAP). RFC 7252, June 2014.

[6] Juraj Somorovsky. Systematic fuzzing and testing of TLS libraries. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pages 1492–1504, New York, NY, USA, 2016. ACM.