Title of the MsC Thesis

Name of author
author.name@ist.utl.pt

Instituto Superior Técnico, Lisboa, Portugal

December 2011

Abstract

Place abstract here. No paragraph breaks.

Keywords: Keyword1, Keyword2, Keyword3, Keyword4, Keyword5

1. Introduction

In recent years there has been a sharp increase in the number of iot devices and this trend is expected to continue[?]. The IoT is a network of interconnected devices, which exchange data with one another over the internet. In fact, it can be any object that has an assigned IP address and is provided with the ability to transfer data over a network. While there are many types of IoT devices, all of them are restricted: they have limited memory, processing power and available energy. This doest not mean, however, that such devices are only capable of running the least demanding algorithms. Various devices, with different hardware characteristics fall under the definition of iot. While for some of them symmetric cryptography is the only viable option, others have resources that allow them to use public key cryptography. Examples of IoT devices include temperature sensors, smart light bulbs and physical activity trackers.

While inter-device communication has numerous benefits, it is important to ensure the security of that communication. For example, when you log in to your online banking account, you do not want others to be able to see your password, as this may lead to the compromise of your account. Having your account compromised means that a malicious entity might take a hold of your money. Despite of all of the benefits that the iot technology brings, communication security is often an afterthought and is frequently ignored.

tls is one of the most used protocols for communication security. It powers numerous technologies, such as https. TLS offers the security services of authentication, confidentiality, privacy, integrity, replay protection and perfect forward secrecy. It is not a requirement to use all of those services for every TLS connection. The protocol is similar to a framework, in the sense that you can enable individual security services on a per-connection basis. Foregoing unnecessary services will lead to a smaller resource usage, which in turn leads to smaller execution time and power usage. This is especially important in the context of IoT, due to the constrained nature of the devices. For example, while confidentiality, integrity and authentication are important when the device communicates with an external service, the first security property is not crucial when the device is downloading a firmware update. In the latter case, integrity and authentication would be enough.

While tls was not designed for the constrained environment of iot, it is a malleable protocol and can be configured to one's needs. For each security service that the protcol offers, there is an array of algorithms that can be used to implement it. If those algorithms are chosen properly, it is possible to use tls in the context of iot.

The majority of existing work on tlsd optimization proposes a solution that is either tied to a specific protocol, such as coap, or requires an introduction of a third-party entity, such as the trust anchor in the case of the S3K system[?] or even both. This has two main issues. First, a protocol-specific solution cannot be easily used in an environment where (D)tls is not used with that protocol. Second, the requirement of a third-party introduces additional cost and complexity, which will be a big resistance factor in adopting the technology. This is especially true for developers working on personal projects or projects for small businesses, leaving the communications insecure in the worse case scenario. Therefore a solution that is protocol independent and fully compatible with the tlsd standard and existing infrastructure is desired.

Another area that the existing literature fails to address is that it almost exclusively focuses on dtls optimization and not all of it can be applied to tls.

Herein we want to further explore tls optimization. There is clearly a need for that, especially with coap over TCP and tls standard[?] being currently developed. The aforementioned standard does not explore any tls optimizations, and since any iot device using it in the future would benefit from them, this is an important area to explore.

The objective of this work is to provide a means of assisting application developers who wish to include secure communications in their applications to make security/resource usage trade-offs, according to the environment's needs and limitations. We aim to provide a general overview of of the costs of the tls protocol as a whole and of its individual parts. This is will allow to answer questions such as *How much will we save if we use protocol X instead of Y for authentication?.* Thus, performing evaluations on specific iot hardware or analyzing tls-specific optimizations on it is outside of the scope of this paper.

In order to achieve our goals, a detailed cost evaluation of tls is needed. With this information, the programer will be able to choose a configuration that meets his security requirements and device constraints. If the limitations of the device's hardware do not allow to meet the requirements, the programer may decide on an alternative configuration, possibly with a loss of some security services and a lower security level, or forgo using (D)tls altogether. Thus, this work is targeted towards developers and InfoSec professionals who wish to add communication security to applications in the IoT environment.

In our work, we performed a thorough cost evaluation of the tls 1.2 implementation in *mbedTLS 2.7.0*. *mbedTLS* is among the most popular tls implementation libraries for embedded systems. We evaluated costs in terms of the estimated number of CPU cycles and time taken. The time values were read directly from the processor's registers. In our analysis we will show that the estimates do reflect real values, by comparing them to time measurements obtained directly from the CPU registers. We evaluated every single one of the 161 tls configurations available in *mbedTLS 2.7.0*, at 4 different security levels.

A tls connection consists of two main parts: first, the peers establish a secure communication channel in the Handshake phase, followed by the data exchange using that channel in the Record phase. We focused on the Handshake part of the protocol for two main reasons. First, it is the part with the most variability in terms of cost, due to the complex combinations of different possible algorithms. Second, it is part which has been the least studied by existing work. The Record phase mainly consists in the use symmetric encryption algorithms and hash functions. Their costs has already been thoroughly studied by existing work.

Although our focus was on the Handshake, we also profiled the costs of the symmetric encryption algorithms and hash functions. We concluded with a typical configuration used on the internet, only a few megabytes While the cost of the Handshake might dominate when small amounts of data are transmitted, ? Our conclusions were that when the devices are expected to transfer a large amount of data (a few megabytes, for a typical configuration used on the internet), the focus should not be on optimizing the Handshake, but rather the algorithms used to to provide data confidentiality and integrity.

2. Background

Place text here...

2.1. Sub-section...

A generic CFD design problem can be formally described as

$$\begin{array}{rl} \text{Minimize} & Y(\alpha, \mathbf{q}(\alpha)) \\ \text{w.r.t.} & \alpha\,, \\ \text{subject to} & \mathcal{R}(\alpha, \mathbf{q}(\alpha)) = 0 \\ & C(\alpha, \mathbf{q}(\alpha)) = 0\,, \end{array} \tag{1}$$

where Y is the cost function, α is the vector of design variables and $\mathbf{q}$ is the flow solution, which is typically of function of the design variables, and $C = 0$ represents additional constraints that may or may not involve the flow solution. The flow governing equations expressed in the form $\mathcal{R} = 0$ also appear as a constraint, as the solution $\mathbf{q}$ must always obey the flow physics.

2.2. Sub-section...

More text...

3. Implementation

Place text here...

3.1. Sub-section...

More text...

3.2. Sub-section...

More text...

4. Results

Place text here...

4.1. Sub-section...

More text...

Figure **??** shows the contour of pressure on the hub and blade surface planes corresponding to the baseline blade geometry.

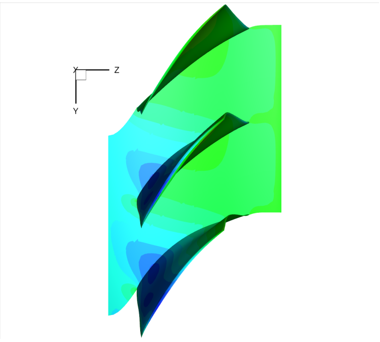

Figure 1: Pressure distribution.

As seen in Fig.**??**...

4.2. Sub-section...

More text...

Table **??** summarizes...

Model	C_L	C_D	C_{My}
Euler	0.083	0.021	-0.110
Navier–Stokes	0.078	0.023	-0.101

Table 1: Table caption

As seen in Tab.**??**...

5. Conclusions

Conclusions, future work and some final remarks...

Acknowledgements

The author would like to thank ...