CoRE Working Group
Internet-Draft
Intended Status: Informational
Expires: August 18, 2014
L. Seitz
SICS Swedish ICT
G. Selander
Ericsson

February 14, 2014

Design Considerations for Security Protocols in Constrained Environments
draft-seitz-ace-design-considerations-00

Abstract

Considerable effort has been spent on securing existing Internet standard authentication and authorization protocols such as TLS, Kerberos, and OAuth, among others. It would save a lot of effort if these protocols could be profiled to be feasible for constrained environments, with some easily obtainable security considerations.

However, these protocols were typically not designed with constrained environments in mind, so profiling of an existing protocol may result in a far from optimal solution. Moreover they are not necessarily complying with their original design objectives outside their intended domain of application.

This document examines the impact of typical characteristics of security protocols (e.g. cryptographic calculations, number and size of protocol messages) in a constrained environment. The goal is to provide decision support when different resource usage optimizations are possible in the adaptation of a security protocol for this setting.

Status of this Memo

The list of current Internet-Drafts can be accessed at http://www.ietf.org/1id-abstracts.html

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

Copyright and License Notice

Table of Contents

1. Introduction

When adapting security protocols for constrained nodes, one has to take into account the various resource limitations. While it might be tempting to optimize the usage of a certain resource (e.g. minimizing RAM consumption), such an approach might produce a less-than-optimal overall solution, compared to a more holistic approach that leverages the combined effect of different optimization possibilities.

The goal of this document is to summarize some characteristics of security protocols and weigh their impact against each other in order to allow effective trade-offs when adapting existing protocols to a constrained setting. While there is some overlap with the scope of the Lightweight Implementation Guidance WG, this document is aimed more at security protocol profiling and design than actual implementation decisions that are the main focus of LWIG.

1.1 Terminology

Certain security-related terms are to be understood in the sense defined in [RFC4949]. These terms include, but are not limited to, "authentication", "authorization", "confidentiality", "encryption", "data integrity", "message authentication code", and "verify".

Terminology for constrained environments is defined in [I-D.ietf-lwig-terminology] e.g. "constrained device".

2. Background

We are assuming a multi-party protocol setting with at least the following parties

a) a resource server hosting resources
b) a client seeking access to some resource, and
c) an authorization server acting Trusted Third Party (TTP) for key distribution and access control handling.

The resource server and/or the client is assumed to be constrained, but the authorization server is not.

The authorization server can provide authentication and authorization means (e.g. cryptographic keys, access control information, certificates) for the other parties.

There are various authentications and authorizations taking place in this multi-party protocol. For example, the client and authorization server mutually authenticates and and the client is being authorized by the authorization server. The resource server needs authenticate information provided by the authorization server, based on a previously established relationship (e.g. shared symmetric keys). Finally when the client communicates with the resource server, the client's authorization needs to be verified, which might include authentication of the client.

Note: Security protocols designed to handle authentication and authorization between two mutually unknown less-constrained peers are not necessarily adapted to the current setting, where optimizations can be made by relying on an relatively unconstrained TTP.

2.1. Device assumptions

Devices may be constrained in different ways, as described in the LWIG terminology document [I-D.ietf-lwig-terminology]. This work is targeting class 1 devices, but may be applicable even the most constrained class of devices (C0) if supported by relevant proxy functionality. Class 2 devices probably do not need any special considerations, since they can mostly support the same protocols as unconstrained devices.

A device for which these considerations apply could e.g. run the following protocol stack, potentially supported by a proxy:

- o The application layer protocol is CoAP [I-D.ietf-core-coap], using UDP at the transport layer.
- o CoAP will be running on top of DTLS [RFC6347].
- o IPv6 [RFC4291] is assumed to be the Internet layer protocol on top of the adaptation layer 6LoWPAN [RFC4944].
- o IEEE 802.15.4 [IEEE802] is assumed as the Link layer protocol for wireless communication. We assume that a large proportion of the target devices will communicate over wireless channels.

2.2 Relevant Factors

From the LWIG terminology draft [I-D.ietf-lwig-terminology] we can list the following resources that need to be considered in general:

- o RAM memory (required state and buffers for running protocols)
- o Flash/ROM memory (required libraries and code complexity)
- o Computational power (required processing speed)
- o Electrical energy (battery consumption, if not mains-powered)
- o User interface and physical accessibility (for performing manual operations directly on the device)
- o Network (bit rate, loss rate, dynamic topology, fragmentation,

lack of advanced services)

The consumers of these resources in the case of security protocols can be summarized as follows:

- o Cryptographic algorithms
  - based on symmetric cryptography
  - based on asymmetric cryptography
  - (orthogonal) implemented by a co-processor (e.g. AES, SHA, ECC)
- o Composing/parsing protocol messages (e.g. Base64 en/decoding, JSON, ASN.1, CBOR)
- o Sending/receiving protocol messages
- o Listening, while waiting to receive protocol messages

2.3 Security protocols in constrained environments

One of the potential advantages with extending basic Internet Protocols to constrained nodes is that other standardized protocols can be applied too.

In particular in the case of security protocols, there is a considerable effort spent to eliminate flaws and weaknesses that could otherwise be exploited for attacking the system. It would save a lot of effort if it was possible to profile these protocols for running efficiently in a constrained environment while maintaining their security properties.

However, the profiling of a protocol may result in a far from optimal solution. For example assume that a constrained profile of a security protocol is made by reducing the message sizes. Such a protocol may still be badly suited for constrained devices e.g. because the number of round trips is what makes the latency high, and reducing that would essentially change the security properties of the protocol.

Moreover, as many of these protocols were not designed for a constrained environment, they are not necessarily complying with their original design objectives outside their intended domain of application. Even security objectives that applied to the Internet may be violated: e.g. a DoS mitigation measure that is based on a processing commitment by a client (a "puzzle", see e.g. [RFC5201]) may be inappropriate if the server is much more constrained than the client.

This memo is intended to support the adaptation of an existing security protocol for a constrained environment by providing some considerations on resource consumption. Furthermore this memo documents the assumptions that were made as a basis for these considerations.

3. Protocol design considerations

3.1 Straightforward optimizations

This section lists some potential targets for resource optimizations.

3.1.1 Smaller messages

Reducing message size will reduce composing/parsing and sending/receiving costs which is favorably impacting energy consumption and latency. Some specific considerations:

- o Smaller than CoAP payload size (1024 bytes) avoids fragmentation at the application layer.
- o Smaller than the maximum MAC-layer frame size (e.g. 127 bytes for IEEE 802.15.4) avoids fragmentation at the link layer.
- o The largest messages are potentially those containing certificates or authorization tokens, so reducing their size significantly will have a large impact.

3.1.2 Fewer messages

Removing message exchanges or round trips have potentially large impact on energy consumption and latency.

Reducing the number of messages in a given security protocol is in general not possible without changing the essential security properties of the protocol. Experiments by Google with TLS false start [I-D.bmoeller-tls-falsestart] and TLS snap start [I-D.agl-tls-snapstart] illustrate the difficulty of trying to reduce the number of messages in an established security protocol.

Challenge-response based authentication protocols may potentially be replaced with other protocols with alternative measures to ensure freshness, such as time or sequence numbers. Such an approach would require fewer message passes, but ensuring freshness can be problematic, since some constrained devices may not be able to reliably measure time.

On the other hand, there are long lifetime battery powered IEEE 802.15.4e devices implementing Time Slotted Channel Hopping (TSCH) which has good time synchronization properties, since that is required for communication.

3.1.3 Less computations

One way of reducing the complexity of required computations is to reduce the number of public key operations used during normal operations, e.g. by keeping existing sessions alive, or generating session resumption state on a less constrained device. The drawback in this case is that either more RAM or more sending and receiving of messages are needed.

An alternative is to replace public key operations with symmetric key operations. Significant reductions in resource consumption can be achieved by using symmetric cryptography instead of asymmetric cryptography, since asymmetric cryptography generally requires larger libraries (e.g. BigInteger, elliptic curves), and consumes more RAM, processing power and energy than symmetric algorithms.

However, it is not always possible to make this replacement as some of the properties of asymmetric cryptography, such as non-repudiation of signatures, and non-confidential key distribution do not apply for symmetric keys. It may require a change in trust model, where a TTP is assumed e.g. for key management.

3.1.4 Reduce RAM usage

Reducing the usage of RAM memory can be achieved by reducing the size of variable state information required by a protocol. Different security protocols and -modes have different requirements in this respect. Optimizations may potentially be done by profiling certain options of the protocol to predefined, default values.

Another possibility is to simplify parsing and processing of protocol messages, leading to smaller libraries that need to be loaded into memory. Further the size of the protocol messages, e.g. certificates and authorization tokens, directly affects the size of the buffers that need to be allocated for receiving and sending them, so keeping them small also helps.

3.1.5 Reduce code size

The overall size of the code is influenced mainly by the size of the libraries needed for cryptography and parsing messages (ASN.1, JSON, XML). In general asymmetric cryptography requires larger libraries (e.g. BigInteger, Elliptic curves) than symmetric cryptography. Minimal libraries for parsing ASN.1 and JSON are roughly comparable in size (around 6 kB) while even minimal XML parsers generally have a significantly larger size.

3.2 Trade-offs

This section looks at the more difficult question how to weigh different optimizations against each other. We emphasize in this section the potential role of the authorization server as an enabler for some of the optimizations.

3.2.1 Fewer vs smaller messages

When comparing reduction of message size versus sending fewer messages in total, if one takes into account the overhead of setting up a bearer, it is more efficient to send longer messages than shorter messages. Considering fragmentation it is better to send messages shorter than the fragmentation limit. Therefore optimal message size seems to be just below the fragmentation limit. Note that fragmentation carries an additional performance penalty in excess of just adding the overhead of sending several fragments, since fragmenting a message increases the risk that a fragment is lost and that the message as a whole needs to be retransmitted.

3.2.2 Crypto vs message exchange

It is known that in wireless constrained devices, the energy consumption for sending and receiving messages is high, and significantly higher than symmetric crypto operations [Margi10impact] and [Meu08engery]. Hence if it is possible to send fewer messages at the cost of delegating some symmetric crypto to the constrained device, such a trade off is favorable. The potential drawback is increased latency and code size. The latter could probably be avoided by reusing existing symmetric algorithms that are needed anyway.

Results from [Meu08engery] indicate that energy consumption for public key operations is on par or greater than message exchange for a particular security protocol. However, the efficiency of processing is increasing: The processing power follows Moore’s law (up to point) and depends on the manufacturing technology while the transmission/reception power is based on laws of physics laws that don’t change with manufacturing. So processing will be more and more energy efficient (up to a point) while the transmission/reception remains almost stable in terms of energy efficiency.

3.2.3 Transmitting vs receiving messages

Results comparing energy consumption of transmitting versus receiving messages seem contradictory. While [Margi10impact] indicates that receiving a message is much cheaper in energy consumption, than sending, [Meu08engery] seems to suggest that both costs are roughly on par.

An important point from [Meu08engery] is that one should consider the cost of listening for the next message in a protocol, while the other party is performing some computations. It is not obvious how much impact smart listening techniques such as Low Power Listening (LPL) or X-MAC [Bue06xmac] have.

Our conclusion on this issue is that is warrants further investigation in order to determine whether it should influence protocol design and profiling or not.

3.2.4 Distributing costs over deployment life time

Provisioning (e.g. access control lists) has a cost which potentially may be amortized over the lifetime of a deployment. Security protocol establishment (e.g. DTLS handshake) may similarly have a high cost that but can be acceptable, if the established session can be used for a long time. The drawback is that storage or RAM memory is consumed to save the state of the provisioned data or the established protocol.

3.2.5 Outsourcing heavy computations

A method of saving computational effort is to outsource computations to a less constrained TTP e.g. authorization decisions and policy management to the authorization server. Note however that this may be changing the trust model of the original protocol, and if the constrained device needs the result of the outsourced computation, this information must be transported in a secure way which in turn incurs a non-negligible cost.

3.2.6 DoS mitigation and anti-spoofing in the Internet

As we have seen it is important in a wireless constrained environment to restrict the number of messages sent and received in a protocol.

Some Internet security protocols include DoS mitigation or anti-spoofing mechanisms such as cookies (cf. [RFC6347]) or puzzles (cf. [RFC5201]) which adds message size and/or round trips. These mechanisms were in general not designed for a constrained environment and may potentially make the protocol unnecessarily heavy without efficiently providing the desired effect.

In fact the existence of a TTP allows for more efficient mechanisms, e.g. that a client first commits or proves source address to the authorization server which can assert such properties in an authorization token verified by a constrained server.

3.2.7 Outsourcing key management

Securing communication between two mutually unknown less-constrained peers has a high cost in terms of additional round trips, e.g. to protect against requests from spoofed initiators, DoS mitigation, challenge response protocols etc. In addition, both parties are often contributing to the generation of key material, which requires exchange of data used in key generation. These costs are a consequence of the trust model and is clearly not adapted to the current setting, where optimizations can be made by relying on an relatively unconstrained authorization server.

In addition to providing authorization decisions, the authorization server may support authentication and authorization between resource server and client by e.g.

- o providing symmetric keys to support authentication (cf. Kerberos).
- o providing protected assertions containing statements about client and server, including public key certificates.

3.2.8 Verifying authorization

As noted above, it is desirable to verify authorization of a request as early as possible in a protocol, to reduce unnecessary message exchanges and processing. However, if that involves verifying a digital signature, then the operation is in itself heavily resource consuming and would preferably only take place after it is known that the request is authorized. This is obviously a "catch 22" and there are various options to attempt to design around this.

In the present case, where we assume a TTP with a previously established relationship - say a shared symmetric key - with the resource server, the legitimacy of the request may e.g. be indicated with a Message Authentication Code instead of a digital signature over an authorization decision.

Authentication of client and server may still require verification of digital signature if public keys are used. However, as noted above, the authorization server may also support key distribution and provide symmetric keys for authentication (cf. Kerberos).

4. Security Considerations

This memo deals with design considerations for security protocols, including security trade-offs that can be made to save resources, some of which will come at the cost of weakening security.

Since a security protocol itself consume resources, one factor that needs to be taken into consideration is the possibility for attackers to use these very security protocols in order to mount a denial of service attack.

Each profiled or modified security protocol must bear its own security considerations. Protocol designers need to carefully evaluate the feasibility of stronger (and thus more resource consuming) security against the risks incurred by a weaker security that is more easy to implement or execute on a constrained node.

5. IANA Considerations

This document has no actions for IANA.

6. Acknowledgements

The authors would like to thank Sumit Shingal and Vlasios Tsiatsis for contributing to the discussion and giving helpful comments.

7. References

7.1 Normative References

[I-D.ietf-core-coap]
Shelby, Z., Hartke, K., Bormann, C., and B. Frank, "Constrained Application Protocol (CoAP)", draft-ietf-core-coap-18 (work in progress), June 2013.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.

[RFC4944] Montenegro, G., Kushalnagar, N., Hui, J., and D. Culler, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks", RFC 4944, September 2007.

[RFC4949] Shirey, R., "Internet Security Glossary, Version 2", FYI 36, RFC 4949, August 2007.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.

7.2 Informative References

[I-D.ietf-lwig-terminology]
Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained Node Networks", draft-ietf-lwig-terminology-07 (work in progress), February 2014.

[IEEE802]
IEEE Computer Society, "IEEE Standard for Local and metropolitan area networks Part 15.4: Low-Rate Wireless Personal

[I-D.bmoeller-tls-falsestart]
Langley, A., Modadugu, N., and B. Moeller, "Transport Layer Security (TLS) False Start", draft-bmoeller-tls-falsestart-00 (expired draft), June 2010.

[I-D.agl-tls-snapstart]
Langley, A., "Transport Layer Security (TLS) Snap Start", draft-agl-tls-snapstart-00 (expired draft), June 2010.

[Meu08engery]
Meulenaer, G., Gosset ,F., Standaert, F., and L. Vandendorpe, "On the Energy Cost of Communication and

Cryptography in Wireless Sensor Networks", proceedings of the IEEE International Conference on Wireless and Mobile Computing, 2008.

[Margi10impact]
Margi, C., Oliveira, B., Sousa, G., Simplicio, M., Barreto, P., Carvalho, T., Naeslund, M., and R. Gold, "Impact of Operating Systmes on Wireless Sensor Networks (Security) Applications and Testbeds", proceedings of the 19th International Conference on Computer Communications and Networks (ICCCN), 2010.

[Bue06xmac]
Buettner, M., Yee, G., Anderson, E., and R. Han, "X-MAC: A Short Preamble MAC Protocol for Duty-Cycled Wireless Sensor Networks", proceedings of SenSys’06, 2006

[RFC5201] Moskowitz, R., Nikander, P., Jokela, P., Ed., and T. Henderson, "Host Identity Protocol", RFC 5201, April 2008.

Authors’ Addresses

Ludwig Seitz
SICS Swedish ICT AB
Scheelevagen 17
22370 Lund
SWEDEN
EMail: ludwig@sics.se

Goeran Selander
Ericsson
Farogatan 6
16480 Kista
SWEDEN
EMail: goran.selander@ericsson.com