Position Paper for Internet of Things Software Update Workshop (IoTSU)

Russ Housley

4 May 2016

INTRODUCTION

I am the author of RFC 4108 [1], "Using Cryptographic Message Syntax (CMS) to Protect Firmware Packages". The specification took about four years to develop, with about half of the time being spent before the concept was brought to the IETF for standardization. Major milestones were:

Dec. 2001:	My work began
Feb. 2003:	First Internet-Draft posted
Nov. 2004:	IETF Last Call
Jan. 2005:	Final Internet-Draft posted
Aug. 2005:	RFC 4108 published

In RFC 4108, a digital signature on the firmware package is used to preserve integrity and provide data origin authentication. The CMS [2] provides a standard format for that signature as well as attributes for the signed content.

PROTECTING FIRMWARE PACKAGES

Many devices contain more than one hardware component that need compatible firmware. For this reason, firmware packages contain object code for one or more hardware components.

Systems are composed from many diverse devices, and an authority needs to be able to load compatible firmware in all of the devices, often through proxies or cumbersome air-gap interfaces. The features in

RFC 4108 that support this complex environment include:

- A digital signature is to protect firmware packages from undetected modification and provide data origin authentication;
- Encryption is optionally used to protect the firmware package from disclosure; and
- A firmware package loading receipt can optionally be generated to acknowledge the successful loading of a firmware package on a hardware module (identified by unique permanent serial number).

The authority needs to be able to determine the firmware package that is appropriate for each device in the system. A common firmware package format allows the authority to easily determine:

- The hardware module type and revision needed to run the firmware package;
- A firmware package identifier and version number;
- A human-readable description of the firmware package;
- Dependencies on other firmware packages, if any;
- A way to designate a subset of hardware modules (called a community) that are authorized to load the firmware package;
- When the firmware package is encrypted, the identifier of the firmware-decryption key;
- Cryptographic algorithms implemented by the firmware package;
- Compression algorithms implemented by the firmware package;
- The identity of signer of the firmware package;
- The certificate of the signer of the firmware package; and
- The date and time that the firmware package was signed.

It is inevitable that a firmware package with a disastrous flaw is released at some point. Once a fixed version is available, subsequent firmware package versions designate a stale version, and the bootstrap loader should prevent loading of the stale version (and earlier ones).

Some people have been very critical of the use of ASN.1. Indeed, better development tools for ASN.1 are needed. However, the type-length-value nature of ASN.1 distinguished encoding makes it pretty easy to write very tight code to extract specific values from the firmware package. And, in some cases, templates are a straightforward alternative.

Different people like to use very different approaches for firmware package identifiers and very different approach to version numbering.

Sadly, there is not a cross-platform convention that everyone can accept. As a result, all of the various approaches need to be supported in some manner.

HASH-BASED DIGITAL SIGNATURES

The Merkle Tree Signature (MTS) algorithm is one form of hash-based digital signature that can only be used for a fixed number of signatures. The MTS algorithm is described in [3]. The MTS algorithm uses small private and public keys, and it has low computational cost; however, the signatures are quite large.

The low computational cost for signature verification makes the MTS algorithm attractive for small devices. The signature size might be a concern for small devices; however, the large signature value is amortized over the whole firmware package, which is fairly large itself.

The MTS signature algorithm, when used with a secure one-way hash function, is considered to be post-quantum secure. The signature algorithms used to sign firmware packages today are not. This means that the distribution of firmware packages could be compromised if a significant advance is made in factoring large integers or a quantum computer is invented. The use of MTS signatures to protect firmware packages seems to offer a way to preserve security for firmware packages in the face of these advances.

The use of the MTS algorithm in a manner compatible with the CMS and RFC 4108 is described in [4].

SUMMARY

A digital signature on the firmware package is essential to preserve integrity and provide data origin authentication. The Merkle Tree Signature (MTS) algorithm seems like a very good choice that will survive significant advances in factoring or invention of a quantum computer.

A common firmware package format makes it much easier for an authority manage the configuration for a diverse population of devices. RFC 4108 offers one choice for a digitally signed firmware package format.

REFERENCES

[1] https://www.rfc-editor.org/rfc/rfc4108.txt
[2] https://www.rfc-editor.org/rfc/rfc5652.txt
[3] https://www.ietf.org/id/draft-mcgrew-hash-sigs-04.txt
[4] https://www.ietf.org/id/draft-housley-cms-mts-hash-sig-04.txt