RG50xQ&RM5xxQ Series

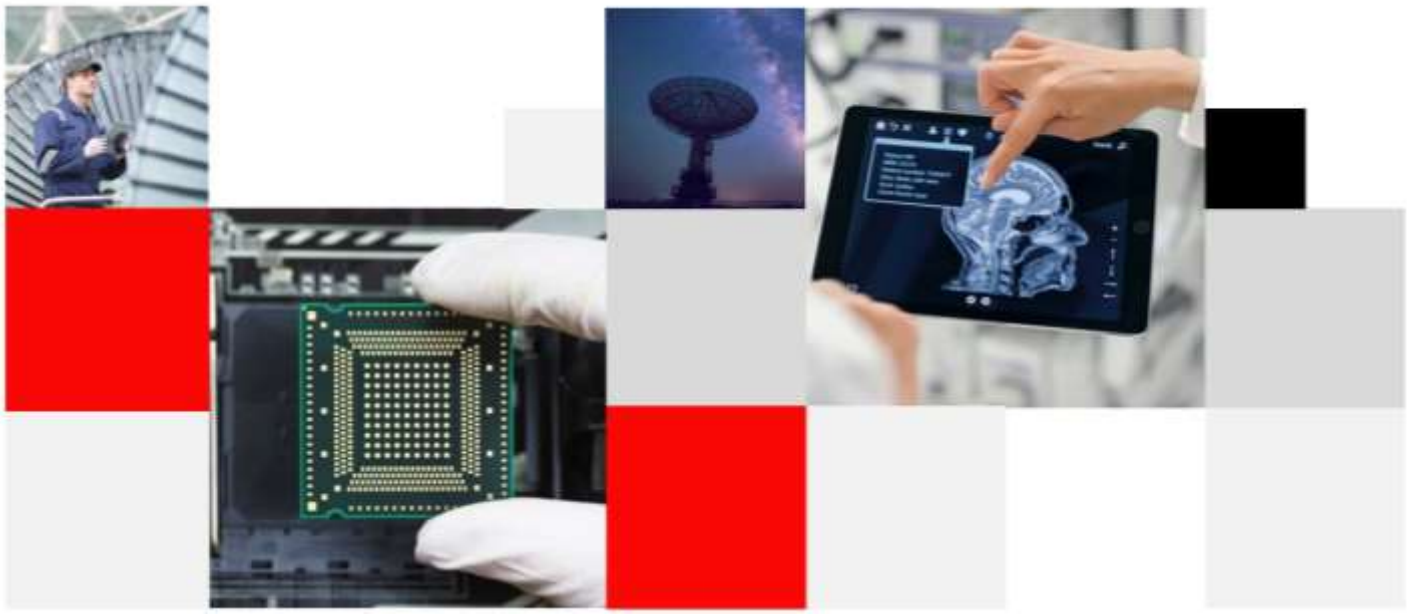
Secure Boot Application Note

5G Module Series

Version: 1.0

Date: 2021-04-02

Status: Released



Build a Smarter World

Our aim is to provide customers with timely and comprehensive service. For any assistance, please contact our company headquarters:

Quectel Wireless Solutions Co., Ltd.
Building 5, Shanghai Business Park Phase III (Area B), No.1016 Tianlin Road, Minhang District, Shanghai 200233, China
Tel: +86 21 5108 6236
Email: info@quectel.com

Or our local office. For more information, please visit:
http://www.quectel.com/support/sales.htm.

For technical support, or to report documentation errors, please visit:
http://www.quectel.com/support/technical.htm
Or email to support@quectel.com.

General Notes

Quectel offers the information as a service to its customers. The information provided is based upon customers' requirements. Quectel makes every effort to ensure the quality of the information it makes available. Quectel does not make any warranty as to the information contained herein, and does not accept any liability for any injury, loss or damage of any kind incurred by use of or reliance upon the information. All information supplied herein is subject to change without prior notice.

Disclaimer

While Quectel has made efforts to ensure that the functions and features under development are free from errors, it is possible that these functions and features could contain errors, inaccuracies and omissions. Unless otherwise provided by valid agreement, Quectel makes no warranties of any kind, implied or express, with respect to the use of features and functions under development. To the maximum extent permitted by law, Quectel excludes all liability for any loss or damage suffered in connection with the use of the functions and features under development, regardless of whether such loss or damage may have been foreseeable.

Duty of Confidentiality

The Receiving Party shall keep confidential all documentation and information provided by Quectel, except when the specific permission has been granted by Quectel. The Receiving Party shall not access or use Quectel's documentation and information for any purpose except as expressly provided herein. Furthermore, the Receiving Party shall not disclose any of the Quectel's documentation and information to any third party without the prior written consent by Quectel. For any noncompliance to the above requirements, unauthorized use, or other illegal or malicious use of the documentation and information, Quectel will reserve the right to take legal action.

Copyright

About the Document

Revision History

Version	Date	Author	Description
-	2021-03-22	Ritchie WU	Creation of the document
1.0	2021-04-02	Ritchie WU	First official release

Contents

Table Index

1 Introduction

Quectel 5G RG50xQ series and RM5xxQ series modules support Secure Boot function. This document describes how to use AT commands to enable the Secure Boot function on RG50xQ series and RM5xxQ series modules, including an overview of Secure Boot, detailed explanations of AT commands, and precautions.

1.1. Applicable Modules

Table 1: Applicable Modules

Module Series	Model
RG50xQ	RG500Q Series
	RG501Q-EU
	RG502Q-EA
RM5xxQ	RM500Q Series
	RM502Q Series
	RM505Q-AE
	RM510Q-GL

2 Secure Boot Overview

2.1. Definition

Secure Boot refers to a secure booting sequence used to establish a trusted platform. Signature verification is additionally required in Secure Boot to ensure that only the verified software can be executed.

At each stage of the module booting process, signature verification needs to be additionally performed during Secure Boot to prevent any software without valid signature or maliciously modified software from running on the module. A root trusted entity is needed during the booting process. Primary Boot Loader (PBL) is embedded in RG50xQ series and RM50xQ series modules as a firmware which is unmodifiable and therefore can serve as the root trusted entity.

2.2. Secure Boot Enabling

Secure Boot function can only be enabled with the fuse on the hardware and cannot be disabled after being enabled.

The module booting process comprises multiple stages. One dedicated image in each stage performs a specific function. After enabling Secure Boot, the image in each stage needs to be verified by the previous image before execution. If the verification cannot be passed, the entire booting process fails, and the module fails to boot up.

As the root of trust, PBL (also known as RoT) is the firmware embedded in chips and therefore cannot be modified. Therefore, the PBL is considered as the most trusted entity in the booting process and used to perform authentication in the next booting stage. Normally, SBL is verified in the second booting stage. The SBL is executed after it has been successfully authenticated by the PBL. Since the SBL has been trusted, the SBL can be used to authenticate the image in the next stage.

2.3. Certificate Chain

Secure Boot supports 2048-bit or 4096-bit RSA private keys for signatures of the certificate and images. The format of the certificate signatures meets the *PKCS v1.2* Standard and the SHA256 or SHA384 algorithm.

The certificate chain of the module supports two-level certificate and three-level certificates. The two-level certificate is adopted by default and consists of two certificates: self-signed root certificate and attestation certificate.

2.4. Image Signing

The standard format of images is ELF. During Secure Boot, to verify each stage of booting process, images of each stage need to be signed. A binary file in the standard ELF format includes several segments indicating different types of information separately, wherein the *hash table segment* stores signature related information. The *hash table segment* also includes the hash values of each segment and information about certificate trust chain.

The images that must be signed in the module are as follows:

- *abl.elf*
- *aop.mbn*
- *devcfg.mbn*
- *hyp.mbn*
- *multi_image.mbn*
- *prog_firehose_sdx55.mbn*
- *sbl1.mbn*
- *tz.mbn*
- *uefi.elf*
- *xbl_cfg.elf*
- *apdp.mbn*
- *NON-HLOS.ubi*

2.5. QFPROM Configuration

RG50xQ series and RM50xQ series modules include one-time programmable fuses. The initial states of all fuses are 0 (which indicates that the Secure Boot function is not enabled). Once a writing operation is performed on the fuse (or the fuse is blown), the state of the fuse permanently becomes 1 (which indicates that the Secure Boot function is enabled). The state cannot be changed after the fuse is blown. To start Secure Boot, the program tool QFPROM is required.

QFPROM is used to store, in NVROM, configurations related to chip authentication and can implement the secure environment required by Secure Boot. Configure QFPROM and then blow the fuse, to complete all security functions such as output of Debug port, JTAG, secure file system and software roolback.

3 Secure Boot Related AT Commands

3.1. AT Command Syntax

3.1.1. Definitions

- **<CR>** Carriage return character.
- **<LF>** Line feed character.
- **<...>** Parameter name. Angle brackets do not appear on the command line.
- **[...]** Optional parameter of a command or an optional part of TA information response. Square brackets do not appear on the command line. When an optional parameter is not given in a command, the new value equals to its previous value or the default settings, unless otherwise specified.
- **Underline** Default setting of a parameter.

3.1.2. AT Command Syntax

All command lines must start with **AT** or **at** and end with **<CR>**. Information responses and result codes always start and end with a carriage return character and a line feed character: **<CR><LF><response><CR><LF>**. Throughout this document, only the commands and responses are presented, while carriage return and line feed characters are deliberately omitted.

Table 2: Types of AT Commands

Command Type	Syntax	Description
Test Command	**AT+<cmd>=?**	Test the existence of corresponding Write Command and return information about the type, value, or range of its parameter.
Read Command	**AT+<cmd>?**	Check the current parameter value of a corresponding Write Command.
Write Command	**AT+<cmd>=<p1>[,<p2>[,<p3>[...]]]**	Set user-definable parameter value.
Execution Command	**AT+<cmd>**	Return a specific information parameter or perform a specific action.

3.2. Declaration of AT Command Examples

The AT command examples in this document are provided to help you familiarize with AT commands and learn how to use them. The examples, however, should not be taken as Quectel's recommendation or suggestions about how you should design a program flow or what status you should set the module into. Sometimes multiple examples may be provided for one AT command. However, this does not mean that there exists a correlation among these examples and that they should be executed in a given sequence.

3.3. AT Commands Description

3.3.1. AT+QSECBOOT Enable or Query Secure Boot

AT+QSECBOOT Enable or Query Secure Boot	
Test Command **AT+QSECBOOT=?**	Response **+QSECBOOT: "status",(**list of supported **<enable>**s**)** **+QSECBOOT: "serialnum",<serial_number>** **+QSECBOOT: "progsec",(**list of supported **<enable>**s**)** **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristic	/

3.3.1.1. AT+QSECBOOT="status" Query Enabling Status of Secure Boot

This command queries whether Secure Boot is enabled in the module.

AT+QSECBOOT="status" Query Enabling Status of Secure Boot	
Write Command **AT+QSECBOOT="status"**	Response **+QSECBOOT: "status",<enable>** **OK** Or **ERROR**
Maximum Response Time	300 ms

Characteristic	/

Parameter

<enable>	Integer type. Enabling status of Secure Boot function. 1 Secure Boot function is enabled 0 Secure Boot function is not enabled

3.3.1.2. AT+QSECBOOT="serialnum" Query the Unique Serial Number of the Module

This command queries the unique serial number of the module.

AT+QSECBOOT="serialnum" Query the Unique Serial Number of the Module	
Write Command **AT+QSECBOOT="serialnum"**	Response **+QSECBOOT: "serilalnum",<serial_number>** **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristic	/

Parameter

<serial_number>	String type. Serial number of the module in hexadecimal format without double quotes.

3.3.1.3. AT+QSECBOOT="progsec" Enable Secure Boot

This command enables Secure Boot function.

AT+QSECBOOT="progsec" Enable Secure Boot	
Write Command **AT+QSECBOOT="progsec"[,<enable>]**	Response If the optional parameter is omitted, query the current setting: **+QSECBOOT: "progsec",<enable>** **OK**

	If the optional parameter is specified, enable Secure Boot function: **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristic	This command takes effect immediately.

Parameter

<enable>	Integer type. Secure Boot function is enabled or not. 1 Enable Secure Boot function 0 Secure Boot function is not enabled (Only valid in the query result.)

4 Matters Needing Attention

1. Secure Boot function can only be enabled with the hardware fuse and cannot be disabled after being enabled.

2. It is recommended to use **AT+QSECBOOT="progsec",<enable>** to enable the Secure Boot function, which is not enabled by default. This command burns the image file in the *sec* partition and automatically activates the Secure Boot function after restarting the module. After the Secure Boot function is enabled, it is not supported to use Firehose to downgrade the firmware to a version that does not support the Secure Boot function.

3. After the Secure Boot function is enabled, it is not supported to downgrade the firmware to a version that does not support the Secure Boot function through DFOTA, otherwise the module cannot be started normally.

4. The PCIe Fuse mode also burns the image file in the *sec* partition, which conflicts with the enabling of Secure Boot. If the Secure Boot function is enabled first, the PCIe Fuse mode cannot be enabled. Thus it is recommended to enable the Secure Boot function of the module after enabling PCIe Fuse mode.

5 Appendix A References

Table 3: Terms and Abbreviations

Abbreviation	Description
DFOTA	Delta Firmware Upgrade Over-The-Air
ELF	Executable and Linkable Format
MBN	Multi Boot Image Format
PBL	Primary Boot Loader
PCIe	Peripheral Component Interconnect Express
PKCS	Public-Key Cryptography Standards
QFPROM	Qualcomm Fuse Programmable Read Only Memory
RPM	RPM Package Manager (originally Red Hat Package Manager)
RoT	Root of Trust
SBL	Secondary Boot Loader
SHA	Secure Hash Algorithm