SecImage Public API Reference

Contents

[Introduction 2](#_Toc421538931)

[System Requirements 2](#_Toc421538932)

[Overview 2](#_Toc421538933)

[SecImage Public API 2](#_Toc421538934)

[Example 2](#_Toc421538935)

[Assumption & Limitation 3](#_Toc421538936)

# Introduction

This documentation is for SecImage public API on how to generate secure images. This allow customers to interface with SecImage feature with function calls (i.e. instead of command line interface)

# System Requirements

1. Python 2.6 and above
2. Windows 7 or Linux
3. Openssl 1.0.1g and above for Linux

# Overview

The SecImage public API provides the capability to:

1. List all need-to-sign images from meta build.
2. Generate image hash from image file.
3. Generate certificates and signature from image hash.
4. Generate signed image with image file and certificate zip package.
5. Validate image.

# SecImage Public API

The SecImage public API is located at **<sectools>/sectools/features/isc/api/intf.py** and can be import as,

from sectools.features.isc.api.intf import SecImageAPIIntf

SecImageAPIIntf.generatehash  
SecImageAPIIntf.generatesigpack  
SecImageAPIIntf.generatesigned

* To enable debug mode (i.e. dump intermediate files in output directory) and increase verbosity (i.e. show debug message)

SecImageAPIIntf().func(args, debug=True, verbose=True)

* To reduce logging,

SecImageAPIIntf().func(args, quiet=True)

NOTE: When quiet mode is set, except error there is no logging from SecImage (i.e. warning is also hidden)

# Example

Example code for client and server side can be found at <Sectools>/example.

Qualcomm Technologies, Inc. is NOT responsible for any misusage of the example codes. OEMs should implement their own client and server infrastructure.

[1] Cient always send the signing attributes in tosign package; and Server is set to accept signing attributes to generate sig package (i.e. ACCEPT\_SIGN\_ATTRS = True & SEND\_SIGN\_ATTRS = True)

[2] Debug mode is enabled, both tosign package and sig package are saved on client workspace.

# Assumption & Limitation

[1] Image hash is calculated with HMAC params which consists of HW\_ID and SW\_ID. Therefore signing options need to be sent to remote signing along with image hash. At remote signer, HMAC params will be checked again and return error if it is not match.

NOTE: attestation cert is always generated at run time (i.e. OU field is created based on signing options). Pre-defined attestation cert is supported by local signer only.

[2] If server set ACCEPT\_SIGN\_ATTRS = Falseand client setSEND\_SIGN\_ATTRS = True, signing request will fail and return error message to prompt client to change the configuration. Server restart is required if this setting changes.

[3] Due to security concern, certificate private keys are not saved even in debug mode.

[4] MRC and 2-level certificate chain are not supported. (i.e. supported by local signer only)