TFNAA01.09
Release Notes

Date of Issue: March 16th, 2012
Project Name: TF Tegra2
Document Type: Technical Specification

Reference & Version: CP-2011-RT-212
Classification: Confidential
Number of pages: 8 (including 1 header pages)

TABLE OF CONTENTS

1 INTRODUCTION

This document contains the *Release Notes* for the Trusted Foundations on Nvidia Tegra2 platform, running Android.
This release is identified by TFNAA01.09.

2 SUPPORTED AND REFERENCE SW COMPONENTS AND TOOLS

This version has been integrated with the Android Gingerbread Nvidia BSP (Linux kernel 2.6.39) and tested on a Tegra2 Ventana board (with “e-fuses” not burnt).

Here are the full details of supported / reference components and tools:

2.1 TF TEGRA2 ANDROID

SW Components & Tools	Purpose	Source	Version
Android BSP and kernel 2.6.39	Android BSP running on Tegra2	Nvidia	`Android ICS for Cardhu.` `BSP version is 14r9`
ARM toolchain	Build of TF driver	Code Sourcery	Gingerbread/prebuilt/linux-x86/toolchain/arm-eabi-4.4.0/bin
ARM toolchain	Build of Normal World Client applications and libraries.	Code Sourcery	Gingerbread /prebuilt/linux-x86/toolchain/arm-eabi-4.4.0/bin
RVCT / RVDS	Build of Secure Services	ARM Limited	4.0 Build 650 (running on Win32)

2.2 WIN32 SIMULATOR

This version has been tested on a PC running Microsoft Windows XP Service Pack 3.

SW Components & Tools	Purpose	Source	Version
Microsoft Visual Studio®	Use of the Windows Simulator	Microsoft	2008 or 2008 Express (running on Win32)

3 NEW FEATURES AND VERSION HISTORY

3.1 NEW FEATURES IN TFNAB01.09

This version:

- Supports a dynamic configuration of UART Secure Trace Driver: UART identifier to use is now dynamically transmitted to the Trusted Foundations inside the TF Boot args structure.
- Extends the maximum supported size used by the Trusted Foundations workspace (maximum size statically extended to 15 MBytes).
- Opens a new API on the L2CC driver to disable/enable the WriteBack and cache linefill options.

3.2 NEW FEATURES IN TFNAB01.08

This Trusted Foundations version adds the support of LP1 Power transition mode on Tegra2 platform.

3.3 NEW FEATURES IN TFNAB01.07

This version introduces the support of boot parameters. Some properties that were previously defined statically at postlink time are now configurable through the boot parameters of the Trusted Foundations (For now, the set of parameters supported is limited to some “hardware” descriptions).

This package also includes a new version of the UART Secure Trace Driver (check UART availability).

3.4 NEW FEATURES IN TFNAA01.06

This version adds:

- An improvement on the Secure Trace Driver (based on UART, robustness improved during LP1),
- The support of Linux OS for the tf_resc and tf_postlinker tools,
- A new version of the tf_postlinker tool: It now support multi-stage postlink (i.e. Postlink step can be repeated several time to aggregate incrementally secure services).

3.5 NEW FEATURES IN TFNAA01.05

This version adds Tag and Latency properties from L2 Cache Controller into the secure world configuration file.

3.6 NEW FEATURES IN TFNAA01.04

Power management framework of Trusted Foundations for Tegra2 has been re-factored to match Linux k39 changes.

3.7 NEW FEATURES IN TFNAA01.03

This version adds to the Trusted Foundations on Nvidia Tegra2:

- the support of hardware acceleration for floating point number (based on Neon),
- optimizations of code in charge of power management transitions (flush of cache L2),
- the secure driver to enable traces from a secure services (based on the UART),

3.8 NEW FEATURES IN TFNAA01.02

This release is the complete version of the Trusted Foundations product on the Tegra2 platform.

This release supports:

- a secure integration of the Trusted Foundations to the Tegra2 device leveraging the TrustZone technology to isolate a Trusted Execution Environment and Secure Services from the main operating system,
- all the Trusted Foundations Developer API v3.0,
- the Developer Reference Manual and the Product Reference Manual documentation,
- a reference integration of the Trusted Foundations for the reference boot loader in order to protect and start the Trusted Foundations along with the Secure Boot process of the Tegra2 platform.

3.9 NEW FEATURES IN TFNAA01.01

This is the Beta version of the first Trusted Foundations release on Nvidia Tegra2 with Android.

This release supports the following features:

- Support of the Trusted Foundations Developer API v3.0,

4 RELEASE RESTRICTIONS

4.1 TF TEGRA2 ANDROID

4.1.1 CRYPTOGRAPHIC API

In this release, the maximum size for PKCS11 Object Data (CKO_DATA) that can be stored and retrieved is limited to 4096 bytes.

4.1.2 MEMORY

The total size of the secure services plus TF Core binary is limited to 1020kB, i.e. the complete TF binary with the postlinked Secure Services.

4.2 WIN32 SIMULATOR

4.2.1 CRYPTOGRAPHIC API

In this release, the maximum size for PKCS11 Object Data (CKO_DATA) that can be stored and retrieved is limited to 4096 bytes.

4.2.2 CUSTOM DRIVER

The simulators do not support the Custom Driver features defined in the Trusted Foundations Product Reference Manual.

5 KNOWN ISSUES

[6370] RESET VECTOR OF TEGRA2 PLATFORM IS ALWAYS ACCESSIBLE TO THE NORMAL WORLD OS.

[6370] TRUSTED FOUNDATIONS BOOT CODE IS RUNNING IN UNPROTECTED IRAM DURING LP1 ENTRY/EXIT

IRAM memory is used by the Trusted Foundations during LP1 Power transition mode. This memory doesn't provide any firewall mechanism.

[2618] POSTLINKER TOOL DOES NOT OBEY OBJECT SECTION ALIGNMENT

The postlinker tool, `tf_postlinker.exe`, does not currently link Native services in accordance with the alignment requirements specified in the ELF object file. The current alignment support in the postlinker is fixed at 8-byte alignment, which supports standard C code compiled into ARM and Thumb. Assembler source code which additionally specifies section alignment requirements higher than 8 bytes, such as the following code sample, may not function as expected: `; ALIGN=X requests 2x byte alignment, in this case 32 byte alignment AREA example_alignment_area, CODE, READONLY, ALIGN=5 CODE32 example_function ; Dummy function that does nothing BX lr END` For data the work around for this restriction is to manually copy incorrectly aligned data structures in the code in to dynamic heap memory at the correct alignment, and using an explicit copy. For code, there is no current workaround, as the software marks dynamic heap memory as „execute never" (XN) in the page tables.

[3982] POSTLINKER TOOL DOES NOT ERROR IF STACK SIZE VIOLATES HEAP SIZE

The postlinker tool, `tf_postlinker.exe`, does not currently produce an error if the stack size set for a Secure service or Secure driver would exceed the available heap memory for that component. The Secure World binary will incorrectly postlink; the postlinker produces a binary that can be integrated into a platform. Any use of the component with inadequate heap memory will fail at run-time when it used; in the case of drivers this typically means the system will not boot. Developers must ensure that their component's heap configuration provides adequate space for its memory requirements.

[4984] MAXIMUM SIZE OF CKO_DATA IS LIMITED TO 4KB

CKO_DATA objects up to 4KB can be manipulated with the External Cryptographic API. If you need to store larger objects, you are advised to use the External Secure Storage API instead of the CKO_DATA objects.

[5862] BUFFER SIZE FOR SINGLE OPERATION IS LIMITED TO 1MB

In the external and internal Cryptographic API, the buffer size for single operation is limited to 1MB. The work-around is to use multi-stage (update) operations.