



HiMPP SDK 二次开发网络安全注意事项

文档版本 01
发布日期 2018-12-25

版权所有 © 深圳市海思半导体有限公司 2018。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

商标声明



HISILICON、海思和其他海思商标均为深圳市海思半导体有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受海思公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，海思公司对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

深圳市海思半导体有限公司

地址：深圳市龙岗区坂田华为基地华为电气生产中心 邮编：518129

网址：<http://www.hisilicon.com>

客户服务电话：+86-755-28788858

客户服务传真：+86-755-28357515

客户服务邮箱：support@hisilicon.com



前 言

概述

本文旨在从网络安全的角度，重点分析基于 HiMPP 系列芯片解决方案交付包开发的产品在被使用中，可能面临的与本交付包中 SDK 软件包相关的网络安全的威胁，同时，针对性的给出相应的解决方案。

本文中涉及到的芯片、代码等以 Hi3559AV100 交付包为例进行描述，除非特别说明，对下表所涵盖的芯片均有效。

产品版本

与本文档相对应的产品版本如下。

产品名称	产品版本
Hi3559A	V100
Hi3559C	V100

读者对象

本文档（本指南）主要适用于以下工程师：

- 技术支持工程师
- 软件开发工程师

修订记录

修订记录累积了每次文档更新的说明。最新版本的文档包含以前所有文档版本的更新内容。

文档版本 01 (2018-12-25)

第 1 次正式版本发布



2.5.4 小节涉及修改

文档版本 00B01 (2018-05-15)

第 1 次临时版本发布



目 录

1 简介.....	1
2 HiMPP SDK 二次开发网络安全注意事项.....	2
2.1 uboot 使用注意事项.....	2
2.1.1 串口	2
2.1.2 命令行	4
2.2 Linux 及 Busybox 使用网络安全注意事项	5
2.2.1 Root 账户	5
2.2.2 运行权限	6
2.2.3 Telnet 服务.....	8
2.2.4 Tftp 服务.....	9
2.2.5 文件权限	9
2.2.6 调试工具	9
2.3 linux 驱动使用网络安全注意事项.....	10
2.3.1 串口	10
2.3.2 USB.....	10
2.4 Huawei LiteOS 使用安全注意事项.....	10
2.4.1 串口、虚拟串口及 SHELL	10
2.4.2 NFS、RAMFS、PROC	11
2.5 应用开发安全注意事项.....	11
2.5.1 代码安全	11
2.5.2 Cipher 驱动.....	11
2.5.3 MPI/API 接口	11
2.5.4 SAMPLE/TOOLS/EXTDRV/COMPONENT	12
2.6 其他安全注意事项.....	12
2.6.1 镜像烧写	12
2.6.2 PC 调试工具.....	12
2.6.3 裸片烧写	13
2.6.4 SD 卡 mount 权限	13
2.6.5 JTAG.....	13
3 展望.....	14



1 简介

HiMPP 交付包为芯片解决方案交付包，主要包括芯片资料、硬件资料、SDK 软件包、PC 端工具以及软件资料等。客户可基于此芯片解决方案交付包，开发自定义的各种形态的产品。



2 HiMPP SDK 二次开发网络安全注意事项

2.1 uboot 使用注意事项

2.1.1 串口

串口属于非常通用设备通信的协议，一般是基于 RS232 的串口，用于设备的近端的底层调测。Hi3559AV100 SDK 中的 uboot 缺省打开了串口功能，开发人员可以在 uboot 启动阶段通过按键等方式中断 uboot 的执行过程，进入 uboot 的命令行，执行调试功能。产品正式发布，客户可采用以下措施对串口做出限制：

1. 在正式生产产品中，将串口从物理上删除。
2. 可以删除 uboot 中 uart 初始化相关的代码，达到关闭串口目的。具体方法如下（以 Hi3559AV100 为例）：
 - 1) 修改 u-boot-2016.11/arch/arm/cpu/armv8/hi3559av100/start.S 文件，删除掉 uart_early_init 相关的调用语句：

```
--- a/arch/arm/cpu/armv8/hi3559av100/start.S
+++ b/arch/arm/cpu/armv8/hi3559av100/start.S
@@ -199,9 +199,9 @@ normal_start_flow:
     ldr    x0, =(CONFIG_SYS_INIT_SP_ADDR)
     bic    sp, x0, #0xf                /* 16-byte alignment for ABI compliance */

-    bl     uart_early_init
-    adr     x0, Str_SystemSartup
-    bl     uart_early_puts
+    //bl     uart_early_init
+    //adr     x0, Str_SystemSartup
+    //bl     uart_early_puts
     /* enable I-Cache */
     bl     icache_enable
```

- 2) 修改 u-boot-2016.11/arch/arm/cpu/armv8/hi3559av100/hw_compressed/startup.c 文件，删除掉 uart_early_init 相关调用及 uart_early_puts 代码实现：



```
--- a/arch/arm/cpu/armv8/hi3559av100/hw_compressed/startup.c
+++ b/arch/arm/cpu/armv8/hi3559av100/hw_compressed/startup.c
@@ -32,8 +32,8 @@ extern unsigned char input_data_end[];
 extern unsigned long _armboot_start;

/*****
-#define error(_s)          uart_early_puts(_s)
-#define putstr(_s)         uart_early_puts(_s)
+#define error(_s)          do {} while(1)//uart_early_puts(_s)
+#define putstr(_s)         do {} while(1)//uart_early_puts(_s)

#include "hw_decompress.c"
#define GZIP_SIZE_OFFSET 0x4
@@ -50,8 +50,8 @@ void start_armboot(void)
 unsigned int image_data_len;
 int pdst_len;
 int ret;
 uart_early_init();
 uart_early_puts("\r\nUncompress ");
+ //uart_early_init();
+ //uart_early_puts("\r\nUncompress ");

/*use direct address mode*/
hw_dec_type=0;
@@ -65,12 +65,12 @@ void start_armboot(void)
 image_data_len = input_data_end - input_data;
 pdst_len = *(int *) (input_data_end - GZIP_SIZE_OFFSET);
 ret = hw_dec_decompress(pdst_h32, pdst_l32, &pdst_len, input_data_h32, input_data, image_data_len, NULL);
 if (!ret)
     uart_early_puts("Ok!");
 else {
     uart_early_puts("Fail!");
     while(1);
 }
 //if (!ret)
 //    uart_early_puts("Ok!");
 //else {
 //    uart_early_puts("Fail!");
 //    while(1);
 //}

/*unit hw decompress IP*/
hw_dec_unit();
```

- 3) 修改 u-boot-2016.11\drivers\serial\serial_pl01x.c 文件中 pl01x_putc pl01x_getc pl01x_tstc pl01x_serial_putc pl01x_serial_tstc pl01x_serial_getc pl01x_serial_setbrg pl01x_serial_initialize pl01x_serial_probe pl01x_serial_setbrg pl01x_serial_putc pl01x_serial_pending 的实现代码，将其设置为空函数。



```
--- a/drivers/serial/serial_pl01x.c
+++ b/drivers/serial/serial_pl01x.c
@@ -35,166 +35,39 @@ static struct pl01x_regs *base_regs __attribute__((section(".data")));

static int pl01x_putc(struct pl01x_regs *regs, char c)
{
    /* Wait until there is space in the FIFO */
    if (readl(&regs->fr) & UART_PL01x_FR_TXFF)
        return -EAGAIN;

    /* Send the character */
    writel(c, &regs->dr);

    return 0;
+return 0;
}

static int pl01x_getc(struct pl01x_regs *regs)
{
    unsigned int data;

    /* Wait until there is data in the FIFO */
    if (readl(&regs->fr) & UART_PL01x_FR_RXFE)
        return -EAGAIN;

    data = readl(&regs->dr);

    /* Check for an error flag */
    if (data & 0xFFFFF00) {
        /* Clear the error */
        writel(0xFFFFFFFF, &regs->ecr);
        return -1;
    }

    return (int) data;
+return 1;
}

static int pl01x_tstc(struct pl01x_regs *regs)
{
    WATCHDOG RESET();
    return !(readl(&regs->fr) & UART_PL01x_FR_RXFE);
+return 1;
}
```

- 4) 修改./u-boot-2016.11/common/board_r.c 文件，在 run_main_loop 中实现 linux 镜像加载（客户可以参考快速启动 sample u-boot 中的 app.c 文件进行实现）。

2.1.2 命令行

Uboot 下的命令行有许多供开发人员开发和调试使用的命令，在正式产品中是不需要用的。

客户只保留产品必需使用的命令：bootm、go、sf(nand/mmc/ufs)、setenv、saveenv。

其他命令可采用修改 uboot 的代码，去掉 uboot 下的开发和使用的命令，修改方式如下：

在 uboot 源代码中，根据工具名称找到对应的源代码文件，然后再 Makefile 中删除对工具源码文件进行编译的语句，下面以 mw 工具为例进行详细说明。

1. 根据 mw 工具的说明信息“mw - memory write (fill)”，查找实现 mw 工具的源代码文件。
从查找到的信息中，可以确认 mw 工具的源文件为 cmd/mem.c。
2. 在 cmd/Makefile 文件中注释掉 COBJS-\$(CONFIG_CMD_MEMORY) += mem.o，或者删除 CONFIG_CMD_MEMORY 宏的定义，即不编译 mw 工具。



其他命令的删除方法，与以上操作类似。

2.2 Linux 及 Busybox 使用网络安全注意事项

2.2.1 Root 账户

Hi3559AV100 SDK 中的 Busybox 缺省只有 Root 用户，没有设置密码。仅供开发人员开发调试使用，实际产品中客户需要对 Root 账户进行加固，以下措施供客户参考。

设置 root 账户密码

步骤 1. 设置登录需要输入账户和密码

修改 busybox 的 ./loginutils/getty.c 文件，设置登录需要输入账户和密码功能，具体方法如下：（左侧为代码登录时，不需要输入账户和密码；右侧代码登录时需要输入账户和密码。）

```
diff --git a/loginutils/getty.c b/loginutils/getty.c
index f6ae3bb..908f387 100644
--- a/loginutils/getty.c
+++ b/loginutils/getty.c
@@ -705,8 +705,8 @@ int getty_main(int argc UNUSED_PARAM, char **argv)
     }
 }

- /*logname = NULL;*/
- logname = "root";
+ logname = NULL;
+ /*logname = "root";*/
     if (!(option_mask32 & F_NOPROMPT)) {
         /* NB: init_tty_attrs already set line speed
          * to G.speeds[0] */
@@ -734,7 +734,7 @@ int getty_main(int argc UNUSED_PARAM, char **argv)
     /* We use PATH because we trust that root doesn't set "bad" PATH,
     * and getty is not suid-root applet */
     /* With -n, logname == NULL, and login will ask for username instead */
- /*BB_EXECLP(G.login, G.login, "-", logname, (char *)0);*/
- BB_EXECLP(G.login, G.login, "-f", logname, (char *)0);
+ BB_EXECLP(G.login, G.login, "-", logname, (char *)0);
+ /*BB_EXECLP(G.login, G.login, "-f", logname, (char *)0);*/
     bb_error_msg_and_die("can't execute '%s'", G.login);
 }
```

步骤 2. 修改/etc/passwd 和/etc/shadow 文件

root 账户默认密码设置为空，修改方式如下：

- 修改/etc/passwd 文件内容为：
root:x:0:0:root:/root:/bin/sh
- 添加/etc/shadow 文件，/etc/shadow 文件的内容设置为：
root:yf/wq7vpRPGxE:0:0:99999:7:::

设置完成后，启动单板时，需使用密码登录。可以在系统启动之后使用 passwd 命令修改默认密码，修改后的密码以加密数据的形式保存在/etc/passwd 和/etc/shadow 文件中。



步骤 3. 禁止 shell 登录

要在系统启动后，禁止 shell 登录，需要执行以下操作：

- 修改/etc/passwd 配置文件

将/etc/passwd 文件中的内容：

```
root:x:0:0:root:/root:/bin/sh
```

修改为：

```
root:x:0:0:root:/root:/bin/false
```

- 修改/etc/shadow 配置文件

将/etc/shadow 配置文件中的内容：

```
root:yf/wq7vpRPGxE:0:0:99999:7:::
```

修改为

```
root!:10000:0:99999:7:::
```

- 修改/etc/inittab 配置文件

将/etc/inittab 文件中，类似：

```
::respawn:/sbin/getty -L ttyS000 115200 vt100 -I "Auto login as I..."
```

的语句删除掉。

----结束

通过以上 3 个步骤的操作后，系统将无法通过 shell 登录单板。

2.2.2 运行权限

建议客户的应用程序以非 Root 方式运行。使用非 Root 方式运行程序需要执行以下操作：

步骤 1. 文件系统需要增加对扩展特性的支持

在执行 `make ARCH=arm64 CROSS_COMPILE=aarch64-himix100-linux- menuconfig` 命令后，在配置界面选择如下配置：

- 对于 jffs2

```
File systems --->
```

```
<*>Journalling Flash File System v2 (JFFS2) support
```

```
[*]JFFS2 XATTR support
```

- 对于 yaffs2

```
File systems --->
```

```
<*>yaffs2 file system support
```

```
[*]Enable yaffs2 xattr support
```

- 对于 ramfs

```
File systems --->
```

```
Pseudo filesystems --->
```

```
[*]Tmpfs extended attributes
```



步骤 2. 增加 filecap 工具，支持普通用户的特权操作

1. 源码包

源码包使用 libcap-ng-x.x.x.tar.gz，客户可在网上下载最新版本。

2. 交叉编译方法

```
$ ./configure --prefix=out --host=aarch64-himix100-linux
$ make && make install
```

3. 编译完成后，将 out/bin/下的可执行文件拷贝到 rootfs/bin 目录下，将 out/lib/下的库文件拷贝到 rootfs/lib 目录下。下面以非 Root 用户访问/dev/mem 设备为例，说明如何为非 Root 用户添加特权属性：

1) 以 Root 用户登录系统。

2) 添加一个新用户 test:

```
~ # adduser test
Changing password for test
New password: *****
Retype password: *****
Password for test changed by root
```

3) 查看/dev/mem 的权限

```
~ # ls -l /dev/mem
crw-rw-rw- 1 root root 1, 1 Jan 1 00:00 /dev/mem
~ #
```

说明只有 root 才有读写权限。

4) 使用 test 用户执行测试程序 btools

```
~ # su test
sh: using fallback suid method
~ $ btools 0x82000000
*** Board tools : ver0.0.1_20121120 ***
====dump memory 0x82000000====
[error]: memmap():open /dev/mem error!
[error]: Memory Map error. exit:0xFFFFFFFF.{source/tools/himd.c:99}
[END]
```

5) 切回 Root 用户，用 filecap 为/bin/btools 添加 sys_rawio 和 sys_admin 权限

```
~ $ su root
# filecap /bin/btools sys_rawio sys_admin
```

再次执行：

```
# filecap /bin/btools
file capabilities
/bin/btools sys_rawio sys_admin
```

此时已经打印 “/bin/btools sys_rawio sys_admin”，说明 btools 的 sys_rawio & sys_admin 能力已经成功添加。

6) 使用 test 用户再次执行测试程序 btools



```
# su test
sh: using fallback suid method
~ $ btools 0x82000000
*** Board tools : ver0.0.1_20121120 ***
====dump memory 0x82000000====
0000: 00000000 2f406567 69766564 2f736563
0010: 2f636f73 3a636f73 61626d61 3132312f
0020: 30303534 69702e30 00736972 49544341
0030: 633d4e4f 676e6168 45440065 54415056
0040: 642f3d48 63697665 732f7365 732f636f
0050: 613a636f 2f61626d 34313231 30303035
0060: 7269702e 53007369 59534255 4d455453
0070: 616c703d 726f6674 464f006d 4d414e5f
0080: 69703d45 00736972 465f464f 4e4c4c55
0090: 3d454d41 636f732f 626d612f 69702f61
00a0: 40736972 34313231 30303035 5f464f00
00b0: 504d4f43 42495441 305f454c 7369683d
00c0: 63696c69 702c6e6f 73697269 5f464f00
00d0: 504d4f43 42495441 4e5f454c 4d00313d
00e0: 4c41444f 3d534149 4e3a666f 69726970
00f0: 4e3c5473 3e4c4c55 73696843 63696c69
[END]
```

打印出上述信息，说明 test 用户使用 btools 工具可以正常打开/dev/mem 设备。

2.2.3 Telnet 服务

telnet 是不安全的传输协议，密码是以明文方式传输，非常容易被网络监听窃取。

Busybox 中的 telnet 服务，仅可用于开发调试，不可用于正式产品中。

客户必须采用以下措施关闭 telnet，具体方法如下，以 Hi3559AV100 为例：

步骤 1. 打开"config_aarch64_a73_a53_softfp_neon"文件。

步骤 2. 查找 telnet 相关的配置选项。

```
820 CONFIG_SLATTACH=y
821 # CONFIG_TCPSVD is not set
822 CONFIG_TELNET=y
823 CONFIG_FEATURE_TELNET_TTYPE=y
824 CONFIG_FEATURE_TELNET_AUTOLOGIN=y
825 CONFIG_TELNETD=y
826 CONFIG_FEATURE_TELNETD_STANDALONE=y
827 CONFIG_FEATURE_TELNETD_INETD_WAIT=y
828 CONFIG_TFTP=y
829 CONFIG_TFTPD=y
830
```



步骤 3. 注释掉 telnet 相关的选项。

步骤 4. 保存文件，并重新压缩 busybox-1.26.2 文件夹为 busybox-1.26.2.tgz。

----结束

2.2.4 Tftp 服务

tftp 是不安全的传输协议，密码是以明文方式传输，非常容易被网络监听窃取。

Busybox 中的 tftp 服务，仅可用于开发调试，不可用于正式产品中。

客户可采用以下措施关闭 tftp。

步骤 1. 打开"config_aarch64_a73_a53_softfp_neon"文件。

步骤 2. 查找 tftp 相关的配置选项。

```
828 CONFIG_TFTP=y
829 CONFIG_TFTPD=y
830
831 #
832 # Common options for tftp/tftpd
833 #
834 CONFIG_FEATURE_TFTP_GET=y
835 CONFIG_FEATURE_TFTP_PUT=y
836 CONFIG_FEATURE_TFTP_BLOCKSIZE=y
837 CONFIG_FEATURE_TFTP_PROGRESS_BAR=y
838 # CONFIG_TFTP_DEBUG is not set
```

步骤 3. 注释掉 tftp 相关的选项。

步骤 4. 保存文件，并重新压缩 busybox-1.26.2 文件夹为 busybox-1.26.2.tgz。

----结束

2.2.5 文件权限

建议客户对 rootfs 中的文件和设备权限按照最小权限的原则进行控制。

2.2.6 调试工具

rootfs 中/bin 目录下的 btools 工具为调试工具，主要是用来调试阶段读写寄存器以及 i2c、spi 接口器件读写命令使用，其中 himc、himd、himd.l、himm、hiddrs、i2c_read、i2c_write、ssp_read、ssp_write 等命令都是 btools 工具的软链接，正式产品中必须删除 btools 工具以及上述软链接。

rootfs 中下所有脚本，均作为 sample 提供，仅供客户或参考或调试或演示。



SDK 包中 OSDRV 目录下还提供了一些单板端调试工具，目录为 osdrv/tools/board。其中 e2fsprogs、gdb、mtd-utils、reg-tools-1.0.0 仅仅作作为开发调试工作交付，不能被用于任何的客户实际的产品中。

2.3 linux 驱动使用网络安全注意事项

2.3.1 串口

串口属于非常通用设备通信的协议，一般是基于 RS232 的串口，用于设备的近端的底层调测。

客户在实际产品中必须采用以下措施来规避风险：

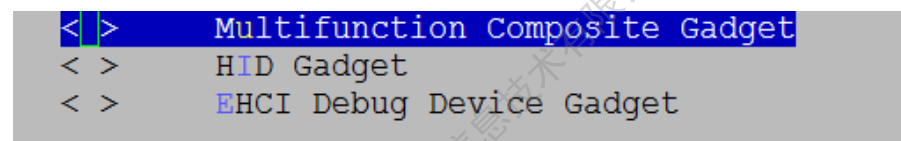
关闭串口：若串口在现网不使用，则在出厂时可以关闭掉串口，这样设备运行中就没有了串口非法接入的风险。

2.3.2 USB

Hi3559AV100 SDK linux 支持 USB 接口的调试接口，客户在实际的产品中必须关闭 USB 接口的调试功能。修改方法如下：

Linux 使用 make menuconfig 界面，不选择“Multifunction Composite Gadget”选项：

```
Device Drivers --->
[*] USB support --->
<*> USB Gadget Support --->
< > Multifunction Composite Gadget
```



```
< > Multifunction Composite Gadget
< > HID Gadget
< > EHCI Debug Device Gadget
```

2.4 Huawei LiteOS 使用安全注意事项

Huawei LiteOS 为华为自主知识产权的操作系统，属于 RTOS 的一种。客户使用 Huawei LiteOS 网络安全注意事项请参考《Huawei LiteOS 安全技术白皮书》。

2.4.1 串口、虚拟串口及 SHELL

串口、虚拟串口、SHELL 仅作为调测使用，客户在实际产品中必须采用以下措施关闭这些功能：

1. 在 Huawei LiteOS 的配置文件.config 中关闭如下选项：

LOSCFG_SHELL

LOSCFG_DRIVERS_UART



2. 代码中禁止调用如下函数：

osShellInit、uart_dev_init、virtual_serial_init、system_console_init、virt_tty_dev_init；

2.4.2 NFS、RAMFS、PROC

NFS、RAMFS、PROC 仅作为调测使用，客户在实际产品中必须采用以下措施关闭这些功能：

在 Huawei LiteOS 的配置文件.config 中关闭如下选项：

LOSCFG_FS_RAMFS

LOSCFG_FS_NFS

LOSCFG_FS_PROC

2.5 应用开发安全注意事项

2.5.1 代码安全

2.5.2 Cipher 驱动

Cipher 驱动实现了标准的对称加解密算法 AES/DES/3DES、非对称加解密、签名验签算法 RSA1024/RSA2048/RSA3072/RSA4096、摘要算法 SHA1/SHA224/SHA256/SHA384/SHA512/HMAC，没有使用任何私有算法。

- DES 算法的安全性低，因为计算能力的提升使得暴力破解成为可能，建议客户不要使用 DES 算法。
- 3DES 算法的安全性低，即使在 K1、K2、K2 三个互不相同的情况下，安全等级也比 AES 128 低，建议客户不要使用 3DES 算法。
- Cipher 密钥的长度越长，安全等级越高。建议客户使用 AES 128bits 及以上的密钥，RSA 2048bits 及以上的密钥。
- SHA1 算法安全性低，不建议客户使用。

2.5.3 MPI/API 接口

- 应用程序在调用需要传入用户路径的 MPI/API 接口时，请应用程序确保传入路径的合法性和正确性。由于 SDK 不提供产品的文件系统分布，建议应用程序调用 realpath 函数，然后进行路径合法性验证。
- 对于 Linux 操作系统，SDK 发布包中的库文件，由于使用了内核的/dev/mem 设备文件，进行物理地址的访问。所以 SDK 发布包的驱动设备节点、MPI、API 等函数被调用时，必须被 Root 用户调用或 user 用户被赋予 CAP_SYS_RAWIO 和 CAP_SYS_ADMIN 文件访问权限的可执行程序使用。
- 对于 Linux 操作系统，SDK 发布包中的 ko 文件，必须被 Root 用户加载/卸载或 user 用户被赋予 CAP_SYS_RAWIO 和 CAP_SYS_ADMIN 文件访问权限的才可加载/卸载。



- SDK 软件包提供给二次开发人员关于芯片对于物理地址的管理、以及配置。对于 SDK 中涉及到物理地址的接口, 请客户保证 MPI、API 接口关于物理地址及长度赋值的正确性, 否则可能导致未知的错误。
- 涉及内存 map/unmap/flush 等操作的接口, 需要用户保证调用顺序正确: 确保 map 后 unmap 前, 方可访问对应内存、进行 flush 等操作, 具体用法参考《HiMPP V4.0 媒体处理软件开发参考.pdf》, 否则可能导致未知的异常。相关接口:
HI_MPI_SYS_Mmap、HI_MPI_SYS_MmapCache
HI_MPI_SYS_Munmap、HI_MPI_SYS_MflushCache
- SDK 驱动支持 select 方式获取码流, 但是 select 实现不支持多线程操作, 需要用户确保使用 select 方式时不使用多线程获取同一个通道的码流。
- SDK 有部分 MPI/API 接口的参数是用户态指针(例如 HI_MPI_VENC_ReleaseStream 等), 需要由用户确保参数是正确值, 否则可能会导致段错误。

2.5.4 SAMPLE/TOOLS/EXTDRV/COMPONENT

SDK 发布包中: sample 目录为示例代码, 支持开发人员快速了解 MPI 接口以及驱动功能; tools 目录为工具目录, 支持开发人员在媒体开发过程中 debug 使用; extdrv 为 Demo 单板配套的外围芯片的驱动软件, 支持 Demo 板演示时使用。

extdrv 为 Demo 单板配套的外围芯片的驱动软件, 部分驱动为第三方提供, 或包含第三方代码, 仅作为 sample 提供, 仅供客户或参考或调试或演示, 具体参考 extdrv 目录下的 readme。

component 目录中的文件, 对客户开源发布, 供客户参考做自定义二次开发, 仅作为 sample 提供。

2.6 其他安全注意事项

2.6.1 镜像烧写

Uboot 下提供的 USB 和 SD 卡烧写功能, 不同于实际产品升级功能。由于镜像, 单板和产品形态资源差异, SDK 版本不提供升级功能, 客户需自行开发, 过程中需要考虑升级安全。

2.6.2 PC 调试工具

SDK 包提供如下 PC 调试组件:

- HiTool: 用于调试烧写板端固件
- PQTool: 用于调试图像质量和效果
- AVS_CALIBRATION: 用于拼接图像效果的标定
- IVE_CLIB: 用于 IVE PC 仿真库
- IVE_Tool: 用于 IVEANN/SVM/CNN 转化
- DPU_Tool: 用于 DPU 校正查找表转化
- NNIE_Tool: 用于 NNIE 编译



- NNIE_Lib: 用于 NNIE PC 仿真库

以上工具都不在网调测，不能被用于任何的客户实际的发布产品中。

SDK 包中 OSDRV 目录下提供一些 PC 端调试及文件系统制作工具，目录为 osdrv/tools/pc。此目录下的工具仅在客户开发阶段使用，且只能运行于 PC 中，不能被用于任何的客户实际的产品中。

2.6.3 裸片烧写

SDK 包提供 USB、SD 卡、串口/网口裸片烧写特性，客户在实际产品中必须关闭这些裸片烧写功能：

- USB、SD 卡可以通过硬件上的设计进行关闭。
- 在软件中关闭 USB、SD 卡裸片烧写功能。

以 Hi3559AV100 为例，在 uboot 代码 ./include/configs/hi3559av100.h 中不要定义 CONFIG_AUTO_UPDATE 宏；删除 board/Hi3559AV100/board.c 文件中的 save_bootdata_to_flash 函数，新的 uboot 将不会被写到 flash 中。

2.6.4 SD 卡 mount 权限

客户开发带可插拔存储介质（如 SD 卡，U 盘）产品形态时，请确保 mount 存储设备文件系统前，加上 -o noexec 选项，以避免第三方程序运行。

2.6.5 JTAG

恶意攻击者通过 JTAG 接口，可以篡改系统的任何配置，恶意破坏系统。

建议客户采用以下措施：

产品出厂时，将 JTAG 接口从物理上删除。



3 展望

HiMPP 产品作为网络上的节点，面临着越来越严峻的安全威胁。HiMPP 产品的安全涉及到 HiMPP 产品的自身及交互的云端、客户端、IPC 等对象，从层次上涉及到管理层、物理层、系统层、网络层、业务层等多个层次。客户有必要基于安全威胁分析采用相对应的安全措施。

以下的一些安全原则可供客户参考：

- 适度的安全
安全设计是基于特定的安全危险场景分析，考虑到性能、成本、业务影响，决策采用最合适的安全措施。
- 最小授权
根据职责的需要，给用户、维护人员、网络单元、程序、进程等授予最小的权限和资源。这样能减少潜在的安全风险。
- 主动协同防御
及时识别恶意攻击源，并在攻击造成显著危害前自动删除恶意用户和网络之间的连接。也可以降低连接的带宽和服务质量，以尽量减少负面影响。
- 纵深防御
纵深防御原则涉及到对威胁的多重防御。例如，当一个防御层不够时，另一个防御层将防止造成一个完整的破坏。